2011

Differential formulation of discontinuous Galerkin and related methods for compressible Euler and Navier-Stokes equations

Haiyang Gao
Iowa State University

Recommended Citation

Gao, Haiyang, "Differential formulation of discontinuous Galerkin and related methods for compressible Euler and Navier-Stokes equations" (2011). *Graduate Theses and Dissertations*. 12248.
https://lib.dr.iastate.edu/etd/12248

Differential formulation of discontinuous Galerkin and related methods for compressible Euler and Navier-Stokes equations

by

Haiyang Gao

A dissertation submitted to the graduate faculty
In partial fulfillment of the requirements for the degree of
DOCTOR OF PHILOSOPHY

Major: Aerospace Engineering

Program of Study Committee:
Z.J. Wang, Major Professor
Paul Durbin
Hui Hu
Hailiang Liu
Alric Rothmayer

Iowa State University
Ames, Iowa
2011

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 4. A Study of Curved Boundary Representations for 2D High-order Euler Solvers

LIST OF TABLES

ABSTRACT

A new approach to high-order accuracy for the numerical solution of conservation laws introduced by Huynh and extended to simplexes by the current work is renamed CPR (correction procedure or collocation penalty via reconstruction). The CPR approach employs the differential form of the equation and accounts for the jumps in flux values at the cell boundaries by a correction procedure. In addition to being simple and economical, it unifies several existing methods including discontinuous Galerkin (DG), staggered grid, spectral volume (SV), and spectral difference (SD).

The approach is then extended to diffusion equation and Navier-Stokes equations. In the discretization of the diffusion terms, the BR2 (Bassi and Rebay), interior penalty, compact DG (CDG), and I-continuous approaches are used. The first three of these approaches, originally derived using the integral formulation, were recast here in the CPR framework, whereas the I-continuous scheme, originally derived for a quadrilateral mesh, was extended to a triangular mesh.

The current work also includes a study of high-order curve boundaries representations. A new boundary representation based on the Bezier curve is then developed and analyzed, which is shown to have several advantages for complicated geometries.

To further enhance the efficiency, the capability of h/p mesh adaptation is developed for the CPR solver. The adaptation is driven by an efficient multi-p a posteriori error estimator. P-adaptation is applied to smooth regions of the flow field while h-adaptation targets the non-smooth regions, identified by accuracy-preserving TVD marker. Several numerical tests are presented to demonstrate the capability of the technique.

CHAPTER 1. General Introduction

Introduction

In today's CFD applications, almost all the production solvers are based on second order methods. These solvers are capable of delivering design-quality solution to a lot of flow problems with reasonable computing cost, thus have made a great impact in many industry fields. However, for certain flow problems, second order schemes are held back by their low accuracy and dissipative nature. Such examples would include the wave propagation in computational aero-acoustics, complex flow structures in large eddy simulation (LES) and direct numerical simulation (DNS) of turbulence flows, and vortex dominated flows such as helicopter vortex blade interaction. High-order methods, on the other hand, show great promise in delivering much better solution quality with comparable computing costs, and therefore have become the popular choice for applications that require high accuracy and low dissipation.

For a high-order method to be applied to a wide variety of real life flow problems, there are several features to be desired. First of all, the method should be robust, most importantly stable. To be applied to complex geometries, the method should be able to handle unstructured meshes, even mixed/hybrid meshes. A compact stencil is also very beneficial, for implicit schemes and parallel computing. Last but not the least, the ideal high-order method should be efficient, preferably in a differential formulation rather than an integral one.

Numerous high-order methods for compressible flows have been developed in the last two decades. Here, we focus only on those that employ a polynomial to approximate the solution in each cell or element, and the polynomials collectively form a function which is discontinuous across cell boundaries. These methods generally feature a compact stencil due to their local solution reconstruction, which is also ideal for use in unstructured meshes. Commonly used methods of this type include discontinuous Galerkin (DG) [3,4,7-9], staggered-grid (SG) [20], spectral volume (SV) [31-35], and spectral difference (SD) [22-24]. Among these, DG and SV are usually formulated via the integral form of the equation, whereas SG and SD, the differential one. Among these methods, DG is arguably the leader due to its guaranteed stability from its weighted residual formulation, while for other methods, a universal framework for stability of all orders of accuracy in arbitrary types of element is still lacking. Although the DG method is mathematically well-defined and straightforward, its implementation might be quite involved due to its integral form.

Huynh (2007) [18] introduced an approach to high-order accuracy called flux reconstruction (FR). The approach solves the equations in differential form. It evaluates the first derivative of a discontinuous piecewise polynomial function by employing the straightforward derivative estimate together with a correction which accounts for the jumps at the interfaces. The FR framework unifies several existing methods: with appropriate correction terms, it recovers DG, SG, SV and SD methods. In the current work, the FR method is extended to 2D and 3D simplexes, and applied to 2D mixed meshes. Initially this method is called the lifting collocation penalty (LCP) method. Later, due to their tight connection, the FR and LCP methods are renamed correction procedures via reconstruction or CPR. The CPR formulation does not involve numerical integrations; the mass matrix inversion is built-in and therefore not needed; as a consequence, the approach is simpler and generally results in schemes more efficient than those by quadrature-based formulations.

The CPR formulation is then extended to diffusion and Navier-Stokes equations, which involves the second derivatives. There are many approaches for the extension. Here, only the ones with a compact stencil are considered. The four schemes of compact stencil employed are BR2 (Bassi and Rebay) [5], compact DG or CDG [25], interior penalty [11,16], and I-continuous (the value and derivative are continuous across the interface) [19]. The BR2 scheme, an improvement of the non-compact BR1 [3], is the first successful approach of this type for the Navier-Stokes equations. The CDG scheme is a modification of the local DG or LDG [10] to obtain compactness for an unstructured mesh. The interior penalty scheme is employed here with a penalty coefficient using correction function [19]. The I-continuous approach is highly accurate for linear problems on a quadrilateral mesh. Nicknamed “poor man’s recovery”, it can be considered as an approximation to the recovery approach of Van Leer and Nomura [27]. In the current work, all the four approaches are cast in the deferential formulation of the CPR method for 2D mixed meshes. The capability of the CPR framework for diffusion and Navier-Stokes equations is demonstrated. Comparisons are also made for the four approaches mentioned above in terms of accuracy and convergence.

Another topic related to the high-order methods is the representation of the curve wall boundaries. As high-order methods become increasingly mature for real world applications, the high-order boundary treatment becomes a critical issue. In mainstream finite volume solvers, the wall boundary of the geometry is usually represented by piecewise straight segments or planar facets. However, this linear boundary representation is far from sufficient for high-order methods. The curvilinear element was introduced by Gordon and Hall [13-14] to solve the problem, which was also used by Kopriva [20] in the staggered-grid method. Both Bassi and Rebay [4], Wang and Liu [36] also addressed the importance of curve boundary representation in their work on high-order methods. Previously, high-order boundary is represented by extra interpolation points added to a boundary face. The current work introduces a new mapping approach based on the Bezier curve, which is able to reconstruct the boundary with a continuous slope, without introducing any sharp corners. Examination of this technique shows its advantage in reducing spurious entropy production for complex geometries.

Adaptive solution techniques are also developed with the CPR method to achieve greater efficiency. H-refinement stands for local mesh subdivision, while p-refinement local polynomial degree enrichment. It has been shown that p-adaptation is superior in regions where the solution is smooth [21], while h-adaptation is preferred in non-smooth regions such as flow discontinuities, where high-order methods become less effective.

Combined h/p-adaptation offers even greater flexibility for complex problems. For an h/p-adaptation scheme, two indicators needs to be computed: a local error indicator to serve as the adaptation criteria, and a local smoothness indicator, used to decide whether h- or p-adaptation should be applied to a certain region. Error indicators are based error estimation approaches, such as gradient or feature based error estimate [6,15], residual-based error estimate [1,2,26], and adjoint-based error estimate [12,17,28-30]. In the present study, a residual-based approach using a multi-p error estimation is adopted as the error indicator. This error indicator involves no ad-hoc parameters. It is also much more efficient to compute compared to the adjoint-based approach. The accuracy-preserving TVD (AP-TVD) marker [37] is employed in the present study as the non-smoothness indicator to identify the non-smooth regions.

Dissertation Organization

The rest of this dissertation is organized as follows.

Chapter 2, “The CPR formulation for conservation laws” is on the derivation of the CPR method and its application on conservation laws including Euler equations for mixed meshes.

Chapter 3, “Differential Formulation of Discontinuous Galerkin and Related Methods for the Navier-Stokes Equations” is a paper submitted to the Journal of Scientific Computing for publication. The paper details the extension the CPR formulation to diffusion and Navier-Stokes equations. I am the primary author of the paper, responsible for most of the work and writing except Section 4 on Fourier analysis.

Chapter 4, “A study of curved boundary representations for 2D high order Euler solvers” is a paper published in the Journal of Scientific Computing. The paper is about the curved boundary representation approach based on Bezier curve. I am the primary author of the paper, responsible for most of the work and writing.

Chapter 5, “A Residual-Based Procedure for hp-Adaptation on 2D Hybrid Meshes” is a paper published in the 49th AIAA ASM conference proceedings. The paper elaborates the development of h/p adaptation procedure with the CPR flow solver. I am the primary author of the paper, responsible for most of the work and writing.

Chapter 6 is devoted to general conclusions.

References

[1] M. Ainsworth, J.T. Oden, A unified approach to a posteriori error estimation using element residual methods, Numer. Math. 65 (1993) 23–50.

[2] T.J. Baker, Mesh adaptation strategies for problems in fluid dynamics. Finite Elem Anal Des 1997;25:243–73.

[3] F. Bassi and S. Rebay, A high-order accurate discontinuous finite element method for the numerical solution of the compressible Navier–Stokes equations. J. Comp. Phys. 1997;131(1):267–79.

[4] F. Bassi and S. Rebay, High-order accurate discontinuous finite element solution of the 2D Euler equations, J. Comput. Phys. 138, 251-285 (1997).

[5] F.Bassi and S. Rebay, GMRES Discontinuous Galerkin Solution of the Compressible Navier-Stokes Equations. In Karniadakis Cockburn and Shu, editors, 208. Springer, Berlin, 2000.

[6] M. J. Berger and P. Collela, Local adaptive mesh refinement for shock hydrodynamics, J. Comput. Phys., vol. 82, pp. 64-84, 1989.

[7] B. Cockburn and C.-W. Shu, TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws II: general framework, Mathematics of Computation 52, 411-435 (1989).

[8] B. Cockburn, S.-Y. Lin and C.-W. Shu, TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws III: one-dimensional systems, J. Comput. Phys. 84, 90-113 (1989).

[9] B. Cockburn and C.-W. Shu, The Runge-Kutta discontinuous Garlerkin method for conservation laws V: multidimensional systems, J. Comput. Phys., 141, 199 - 224, (1998).

[10] B. Cockburn and C.-W. Shu, The local discontinuous Galerkin methods for time-dependent convection diffusion systems, SIAM J. Numer. Anal., 35(1998), pp. 2440-2463.

[11] V. Dolejší, On the discontinuous Galerkin method for numerical solution of the Navier-Stokes equations, Int. J. Numer. Methods Fluids, 45:1083-1106, 2004.

[12] M. B. Giles and N. A. Pierce. Adjoint equations in CFD: Duality, boundary conditions and solution behavior. AIAA paper 97-1850, 1997.

[13] W. J. Gordon and C. A. Hall, "Construction of Curvilinear Co-ordinate Systems and Applications to Mesh Generation," Int. J. Num. Meth. Engng., 7 (1973), pp 461-477.

[14] W. J. Gordon and C. A. Hall, Transfinite element methods and blending function interpolation over curved element domains. Numer. Math. 21(1973), 109-129.

[15] R. Harris and Z.J. Wang, “High-order adaptive quadrature-free spectral volume method on unstructured grids,” Computers & Fluids 38 (2009) 2006–2025.

[16] R. Hartmann and P. Houston, Symmetric interior penalty DG methods for the incompressible Navier-Stokes Equations I: Method formulation. Int. J. Numer. Anal. Model., 1:1-20.

[17] R. Hartmann and P. Houston. Adaptive discontinuous Galerkin finite element methods for the compressible Euler equations. J. Comput. Phys., 183:508–532, 2002.

[18] H.T. Huynh, A flux reconstruction approach to high-order schemes including discontinuous Galerkin methods, AIAA Paper 2007-4079.

[19] H.T. Huynh, A Reconstruction Approach to High-Order Schemes Including Discontinuous Galerkin for Diffusion, AIAA Paper 2009-403.

[20] D.A. Kopriva and J.H. Kolias, A conservative staggered-grid Chebyshev multidomain method for compressible flows, J. Comp. Physics, 125(1996), pp. 244-261.

[21] Y. Li, S. Premansuthan and A. Jameson, Comparison of h- and p-adaptations for spectral difference method, AIAA 2010-4435.

[22] Y. Liu, Vinokur M, and Wang ZJ. "Discontinuous Spectral Difference Method for Conservation Laws on Unstructured Grids," in Proceedings of the 3rd International Conference on Computational Fluid Dynamics, Toronto, Canada, July 12-16 2004.

[23] Y. Liu, Vinokur M, and Wang ZJ. "Discontinuous Spectral Difference Method for Conservation Laws on Unstructured Grids," Journal of Computational Physics Vol. 216, pp. 780-801 (2006).

[24] G. May and A. Jameson, “A spectral difference method for the Euler and Navier-Stokes equations”, AIAA paper No. 2006-304, 2006.

[25] J. Peraire and P.-O. Persson, The compact discontinuous Galerkin (CDG) method for elliptic problems, SIAM J. Sci. Comput., Vol. 30, No. 4 (2008) pp. 1806-1824.

[26] Shih, T.I-P. and Qin, Y., A Posteriori Method for Estimating and Correcting Grid-Induced Errors in CFD Solutions – Part 1: Theory and Method, AIAA Paper 2007-100.

[27] B. van Leer and S. Nomura, Discontinuous Galerkin for Diffusion, AIAA Paper 2005-5108.

[28] D.A. Venditti, D.L. Darmofal, Adjoint error estimation and grid adaptation for functional outputs: Application to quasi-one-dimensional flow. J Comput. Phys. 2000;164:204–27.

[29] D.A. Venditti, D.L. Darmofal, Anisotropic grid adaptation for functional outputs: application to two-dimensional viscous flows, J. of Comp. Phys, 187 (2003) 22-46.

[30] L. Wang and D. Mavriplis, Adjoint-based h-p Adaptive Discontinuous Galerkin Methods for the Compressible Euler Equations, AIAA-2009-0952.

[31] Z.J. Wang, Spectral (Finite) volume method for conservation laws on unstructured grids: basic formulation, J. Comput. Phys. Vol. 178, pp. 210-251, 2002.

[32] Z.J. Wang, High-order methods for the Euler and Navier-Stokes equations on unstructured grids, Journal of Progress in Aerospace Sciences, Vol. 43, pp. 1-47, 2007.

[33] Z.J. Wang and Y. Liu, Spectral (finite) volume method for conservation laws on unstructured grids iii: one-dimensional systems and partition optimization, Journal of Scientific Computing, Vol. 20, No. 1, pp. 137-157 (2004).

[34] Z.J. Wang and Y. Liu, Spectral (finite) volume method for conservation laws on unstructured grids iii: one-dimensional systems and partition optimization, Journal of Scientific Computing, Vol. 20, No. 1, pp. 137-157 (2004).

[35] Z.J. Wang, L. Zhang and Y. Liu, Spectral (Finite) Volume Method for Conservation Laws on Unstructured Grids IV: Extension to Two-Dimensional Euler Equations, Journal of Computational Physics Vol. 194, No. 2, pp. 716-741 (2004).

[36] Z.J. Wang, Y. Liu, Extension of the spectral volume method to high-order boundary represention, J. Comput. Phys. 211 (2006) 154-178.

[37] M. Yang and Z.J. Wang, A parameter-free generalized moment limiter for high-order methods on unstructured grids, AIAA-2009-605.

CHAPTER 2. The CPR Formulation for Conservation Laws

1. Review of the Flux Reconstruction Method

This review presents the essential idea of the flux reconstruction developed by Huynh [4]. Consider the following scalar conservation law

$$\frac{\partial Q}{\partial t}+\frac{\partial F(Q)}{\partial x}=0\,, \tag{1.1}$$

where Q is the state variable and F is the flux. The computational domain *[a, b]* is discretized into N elements, with the *ith* element defined by $V_i \equiv [x_{i-1/2}, x_{i+1/2}]$. Each element can be transformed into the standard element *[-1, 1]* using a linear transformation. The DOFs at the *ith* element are the nodal values of the state variable $Q_{i,j}$ at *k+1* solution points, $x_{i,j}$, *j = 1, ..., k+1*. Then the solution is approximated by the following degree *k* Lagrange interpolation polynomial

$$Q(x)\approx Q_i^h(x)=\sum_{j=1}^{k+1}L_j(x)Q_{i,j}\,, \tag{1.2}$$

where $L_j(x)$ is the Lagrange polynomial or shape function. Given this numerical solution, the flux at every point is well defined, i.e., $F(Q_i^h(x))$. For non-linear conservation laws, $F(Q_i^h(x))$ may not be a polynomial. Instead, $F(Q_i^h(x))$ is approximated by the following degree *k* flux polynomial

$$F_i\ (x)=\sum_{j=1}^{k+1}L_j(x)F(Q_i^h(x_{i,j}))\,. \tag{1.3}$$

Obviously, if the flux is a linear function of *Q*, $F_i\ (x)$ is identical to $F(Q_i^h(x))$. Since we do not explicitly enforce continuity at element interfaces, the state variable is discontinuous across the interfaces. In order to update the DOFs, a new flux function $\hat{F}_i(x)$ is reconstructed, which must satisfy the following criteria:

- $\hat{F}_i(x)$ is a degree *k+1* polynomial, i.e., one order higher than the solution polynomial;

- $\hat{F}_i(x)$ is close to $F_i\ (x)$ in some sense. In other words, some norm of the difference $\left\|\hat{F}_i(x)-F_i\ (x)\right\|$ is minimized;
- At both ends of the element, the flux takes on the value of the Riemann fluxes, i.e.,

$$\hat{F}_i(x_{i-1/2})=\tilde{F}(Q_{i-1}(x_{i-1/2}),Q_i(x_{i-1/2}))\equiv\tilde{F}_{i-1/2}$$
$$\hat{F}_i(x_{i+1/2})=\tilde{F}(Q_i(x_{i+1/2}),Q_{i+1}(x_{i+1/2}))\equiv\tilde{F}_{i+1/2}\,,$$

where $\tilde{F}(Q^-,Q^+)$ is any Riemann flux given the two discontinuous solutions at the left and right of the interface. Once this flux function is found, the DOFs are updated using the following differential equation

$$\frac{\partial Q_{i,j}}{\partial t}+\frac{\partial\hat{F}_i(x_{i,j})}{\partial x}=0\,. \tag{1.4}$$

Obviously the above criteria do not uniquely define $\hat{F}_i(x)$. As a matter of fact, only two conditions are given. We need *k* extra conditions to define the degree *k+1* polynomial $\hat{F}_i(x)$. Using special polynomials such as the Radau and Legendre polynomials, Huynh [4] successfully recovered $\hat{F}_i(x)$ for the DG, SG (or SD/SV) methods, at least for linear conservations laws. The reconstructed flux is first re-written as

$$\hat{F}_i(x)=F_i\ (x)+\sigma_i(x)\,, \tag{1.5}$$

where $\sigma_i(x)$ is a correction flux polynomial, which should be as close as possible to 0. The correction is then further expressed to satisfy the two end conditions

$$\sigma_i(x)=[\tilde{F}_{i-1/2}-F_i\ (x_{i-1/2})]g_L(x)+[\tilde{F}_{i+1/2}-F_i\ (x_{i+1/2})]g_R(x)\,, \tag{1.6}$$

where $g_L(x)$ and $g_R(x)$ are both degree *k+1* polynomials called correction functions, and they satisfy

$$\begin{aligned} g_L(x_{i-1/2})=1,\quad g_L(x_{i+1/2})=0\\ g_R(x_{i-1/2})=0,\quad g_R(x_{i+1/2})=1. \end{aligned} \tag{1.7}$$

Eq. (1.4) then becomes

$$\frac{\partial Q_{i,j}}{\partial t}+\frac{\partial F_i^h(x_{i,j})}{\partial x}+[\tilde{F}_{i-1/2}-F_i\ (x_{i-1/2})]g_L'(x_{i,j})+[\tilde{F}_{i+1/2}-F_i\ (x_{i+1/2})]g_R'(x_{i,j})=0\,. \tag{1.8}$$

Because of symmetry, we only need to consider $g_L(x)$, or simply *g(x)*. It is more convenient to consider the correction function in the standard element $g(\xi)$ on [-1, 1]. Many correction functions were presented in [4], corresponding to different numerical methods. Several schemes are described next.

1. If *g* is the right Radau polynomial, the resulting scheme is actually the DG method!
2. If *g* has Chebyshev-Lobatto points as its interior roots, the resulting scheme is the SG method or the SD/SV method in 1D. The scheme, however, is mildly unstable, which was also found by Van den Abeele et al [8].
3. If *g* has a vanishing derivative at the right boundary as well as at the interior Legendre-Lobatto points, the method results in a remarkably simple, yet stable scheme if one chooses the Legendre-Lobatto points as the solution points since the corrections for the interior solution points vanish! Huynh named this scheme the g_2 scheme.
4. If *g* has the Legendre-Gauss points as its interior roots, the scheme is stable. This suggests that using the Legendre-Gauss points and the two end points as flux points results in a stable SG method in 1D.

As mentioned earlier, the FR method has some remarkable properties. Its DG formulation looks like a pseudo-DG. However, it is identical to the "real" DG formulation! The natural question next is how about simplex elements and other cell types? This is the focus of the current study.

As can be seen above, reconstructing the full $\hat{F}_i(x_{i,j})$ is actually not necessary, since eventually what is needed is the derivative $\partial\hat{F}_i(x_{i,j})/\partial x$, i.e. the last two terms in the left-hand-side of (1.8). Therefore, a more efficient approach is pursued when attempting to extend this method to 2D and 3D simplexes. The following section describes the basic idea.

3. Lifting Collocation Penalty Formulation

3.1 Basic Idea

Rewrite the hyperbolic conservation law as

$$\frac{\partial Q}{\partial t} + \nabla \bullet \vec{F}(Q) = 0, \tag{2.1}$$

where $\vec{F} = (F^x, F^y)$ is the flux vector. Assume that the computational domain is discretized into N non-overlapping triangular elements $\{V_i\}$. Let W be an arbitrary weighting function. The weighted residual form of (2.1) on element V_i can be easily derived by multiplying (2.1) with W and integrating over V_i to obtain

$$\int_{V_i} \left(\frac{\partial Q}{\partial t} + \nabla \bullet \vec{F}(Q) \right) W dV = \int_{V_i} \frac{\partial Q}{\partial t} W dV + \int_{\partial V_i} W \vec{F}(Q) \bullet \vec{n} dS - \int_{V_i} \nabla W \bullet \vec{F}(Q) dV = 0. \tag{2.2}$$

Let Q_i^h be an approximate solution to Q at element i. We assume that the solution belongs to the space of polynomials of degree k or less, i.e., $Q_i^h \in P^k$ (whose dimension is m), within each element without continuity requirement across element interfaces. In addition, we require that the numerical solution Q_i^h must also satisfy (2.2), i.e.,

$$\int_{V_i} \frac{\partial Q_i^h}{\partial t} W dV + \int_{\partial V_i} W \vec{F}(Q_i^h) \bullet \vec{n} dS - \int_{V_i} \nabla W \bullet \vec{F}(Q_i^h) dV = 0. \tag{2.3}$$

A common Riemann flux is used to replace the normal flux to provide element coupling, i.e.,

$$F^n(Q_i^h) \equiv \vec{F}(Q_i^h) \bullet \vec{n} \approx \tilde{F}^n(Q_i^h, Q_{i+}^h, \vec{n}), \tag{2.4}$$

where Q_{i+}^h is the solution outside the current element V_i. Then (2.3) becomes

$$\int_{V_i} \frac{\partial Q_i^h}{\partial t} W dV + \int_{\partial V_i} W \tilde{F}^n(Q_i^h, Q_{i+}^h, \vec{n}) dS - \int_{V_i} \nabla W \bullet \vec{F}(Q_i^h) dV = 0. \tag{2.5}$$

Applying integration by parts to the last term on the LHS of (2.5), we obtain

$$\int_{V_i} \frac{\partial Q_i^h}{\partial t} W dV + \int_{V_i} W \nabla \bullet \vec{F}(Q_i^h) dV + \int_{\partial V_i} W \left[\tilde{F}^n(Q_i^h, Q_{i+}^h, \vec{n}) - F^n(Q_i^h) \right] dS = 0. \tag{2.6}$$

The last term on the left side of (2.6) can be viewed as a penalty term, i.e., penalizing the normal flux differences. Introduce a "correction field" $\delta_i \in P^k$, which is determined from a "lifting operator"

$$\int_{V_i} W \delta_i dV = \int_{\partial V_i} W [\tilde{F}] dS, \tag{2.7}$$

where $[\tilde{F}] = \tilde{F}^n(Q^h, Q_+^h, \vec{n}) - F^n(Q^h)$ is the normal flux difference. Substituting (2.7) into (2.6), we obtain

$$\int_{V_i}\left[\frac{\partial Q_i^h}{\partial t} + \nabla \bullet \vec{F}(Q_i^h) + \delta_i\right] W dV = 0. \tag{2.8}$$

If the flux vector is a linear function of the state variable, then $\vec{\nabla} \bullet \vec{F}(Q_i) \in P^{k-1}$. In this case, the terms inside the square bracket are all elements of P^k. Because the test space is selected to ensure a unique solution, Eq. (2.8) is equivalent to

$$\frac{\partial Q_i^h}{\partial t} + \nabla \bullet \vec{F}(Q_i^h) + \delta_i = 0, \tag{2.9}$$

For non-linear conservation laws, $\vec{\nabla} \bullet \vec{F}(Q_i)$ is usually not an element of $P^k(V_i)$. As a result, we approximate it by its projection onto $P^k(V_i)$, denoted by $\Pi\left(\vec{\nabla} \bullet \vec{F}(Q_i)\right)$. One way to define this projection is

$$\int_{V_i} \Pi\left(\vec{\nabla} \bullet \vec{F}(Q_i)\right) W \mathrm{d}V = \int_{V_i} \vec{\nabla} \bullet \vec{F}(Q_i) W \mathrm{d}V \tag{2.10}$$

Eq. (2.14) then reduces to

$$\frac{\partial Q_i}{\partial t} + \Pi\left(\vec{\nabla} \bullet \vec{F}(Q_i)\right) + \delta_i = 0. \tag{2.11}$$

With the introduction of the correction field δ_i, and a projection of $\vec{\nabla} \bullet \vec{F}(Q_i)$ for non-linear conservation laws, we have reduced the weighted residual formulation to a differential formulation, which involves no integrals.

The performance of this formulation of course hinges on how efficiently the correction field δ can be computed. To get a sense of its form, letting $W = 1$ in (2.7), we obtain

$$\int_{V_i} \delta_i dV = \int_{\partial V_i} [\tilde{F}] dS, \tag{2.12}$$

or

$$\bar{\delta}_i = \frac{1}{|V_i|} \sum_{f \in \partial V_i} \int_f [\tilde{F}] dS, \tag{2.13}$$

$$\bar{\delta}_i = \frac{1}{|V_i|}\sum_{f\in\partial V_i}\int_f [\tilde{F}]dS, \tag{2.13}$$

where $\bar{\delta}$ is the volume averaged δ. We can assume that the flux difference $[\tilde{F}]$ is a degree k polynomial on face *f*, and can be determined based on values of $[\tilde{F}]_{f,l}$ at a set of "flux points" using a Lagrange interpolation, as shown in Figure 1. Then the face integral can be computed exactly using a quadrature formula such as

$$\bar{\delta}_i = \frac{1}{|V_i|}\sum_{f\in\partial V_i}\sum_l w_l[\hat{F}]_{f,l}S_f , \tag{2.14}$$

where w_l is the quadrature weight for the surface integral, and S_f is the area of face *f*. Next let the degrees-of-freedom (DOFs) be the solutions at a set of points $\{\vec{r}_{i,j}\}$, named solution points (SPs), as shown in Figure 1. Then equation (2.9) must be true at the SPs, i.e.,

$$\frac{\partial Q_{i,j}^h}{\partial t} + \Pi_j\left(\nabla\bullet\vec{F}(Q_i^h)\right) + \delta_{i,j} = 0\,, \tag{2.15}$$

where $\Pi_j\left(\nabla\bullet\vec{F}(Q_i^h)\right) = \left[\Pi\left(\nabla\bullet\vec{F}(Q_i^h)\right)\right]_{\vec{r}_{i,j}}$.

Let's examine (2.7) more carefully. If $[\tilde{F}]$ is assumed to be degree k polynomials along the cell faces, and that the triangle has straight face, the correction field δ_i can be computed explicitly in the following form

$$\delta_{i,j} = \frac{1}{|V_i|}\sum_{f\in\partial V_i}\sum_l \alpha_{j,f,l}[\tilde{F}]_{f,l}S_f , \tag{2.16}$$

where $\alpha_{j,f,l}$ are constant coefficients independent of the solution, but dependent on the weighting function *W*. Substituting (2.16) into (2.15) we obtain the following formulation

$$\frac{\partial Q_{i,j}^h}{\partial t} + \Pi_j\left(\nabla\bullet\vec{F}(Q_i^h)\right) + \frac{1}{|V_i|}\sum_{f\in\partial V_i}\sum_l \alpha_{j,f,l}[\tilde{F}]_{f,l}S_f = 0\,. \tag{2.17}$$

Obviously, this is a collocation-penalty formulation. In addition, a lifting operator is used to completely remove the weighting functions, which are used to compute the constant "lifting coefficients" $\alpha_{j,f,l}$.

Next, we focus on how to compute the projection $\Pi_j(\nabla\bullet\vec{F}(Q_i^h))$, a brute force implementation based on Eq.(2.11) requires high-order integral quadrature, and thus is expensive . Two more efficient approaches are presented here.

Approach 1. Lagrange polynomial approach

Based on the solution at an SP, the flux vector at each SP can be computed. Then a degree k Lagrange interpolation polynomial for the flux vector is used to approximate the nonlinear flux vector.

$$\vec{F}(Q_i) = I_k\left(F(Q_i)\right) \equiv \sum_j L_j^{SP}(\vec{r})\vec{F}\left(Q_{i,j}\right) \tag{2.18}$$

where $L_j^{SP}(\vec{r})$ is the Lagrange polynomial based on the solution points $\{\vec{r}_{i,j}\}$. After that, the projection is computed using

$$\Pi\left(\vec{\nabla}\cdot\vec{F}(Q_i)\right) \approx \vec{\nabla}\cdot I_k\left(\vec{F}(Q_i)\right) = \sum_j \vec{\nabla}L_j^{SP}\cdot\vec{F}\left(Q_{i,j}\right) \tag{2.19}$$

In this case $\Pi(\vec{\nabla}\cdot\vec{F}(Q_i))$ is a degree *k-1* polynomial, which also belongs to P^k. Numerical tests indicate a slight loss of accuracy with the LP approach.

Approach 2. Chain rule (CR) approach

It is recognized that the divergence of the flux vector can be computed analytically given the approximate solution by using the chain rule, i.e.

$$\vec{\nabla}\cdot\vec{F}\left(Q_{i,j}\right) = \frac{\partial\vec{F}\left(Q_{i,j}\right)}{\partial Q}\cdot\vec{\nabla}Q_{i,j}\,, \tag{2.20}$$

where $\dfrac{\partial\vec{F}(Q_{i,j})}{\partial Q}$ is the flux Jacobian matrices, which can be computed analytically, then the projection is approximated by the Lagrange interpolation polynomial of the flux vector divergence at the solution points, i.e.

$$\Pi\left(\vec{\nabla}\cdot\vec{F}(Q_i)\right) \approx \sum_j L_j^{SP}(\vec{r})\vec{\nabla}\cdot\vec{F}\left(Q_{i,j}\right). \tag{2.21}$$

Numerical experiments indicate that the CR approach is much more accurate than the LP approach.

Now that the CPR formulation is complete, let’s examine (2.17) a bit further. It is obviously a collocation-like formulation, with penalty-like terms to provide the coupling between elements. However, with a special choice of scheme coefficients $\alpha_{j,f,l}$, we can show that (2.17) is equivalent to the DG method, at least in the linear case. From (2.7), it is easy to show that δ_i is uniquely defined once proper test functions W and the flux points are given. Therefore, the location of the solution points does not affect the solution polynomial $Q_i^h(\vec{r})$ (although $Q_{i,j}^h$ obviously depends on the location of $\vec{r}_{i,j}$) or the correction polynomial. Therefore, the most efficient choice is to make the solution points coincide with the flux points because no data interpolations are then needed for flux computations. Any convergent nodal sets with enough points at the element interface are good candidates, e.g., those found in [1,3,9]. Figure 2 shows the location of the DOFs for k = 1, 2 and 3.

3.2 Connection between the CP and DG, SV and SD Methods

Let's first express the solution and the correction in terms of the DOFs, i.e.,

$$Q_i^h = \sum_j L_j Q_{i,j}^h \,, \tag{2.22}$$

$$\delta_i = \sum_j L_j \delta_{i,j} \,, \tag{2.23}$$

where L_j is the Lagrange polynomial based on the solution points. In the DG method, the weighting function W is set to be one of the Lagrange polynomial L_j. Substituting W into (2.7), we obtain the following equations

$$\int_{V_i} L_k \sum_j L_j \delta_{i,j} \, dV = \sum_{f \in \partial V_i} \int_f L_k \sum_l L_l [\hat{F}]_{f,l} \, dS \,, \;\; k = 1,..., m. \tag{2.24}$$

The unknowns in (2.24) $\delta_{i,j}$ can be easily solved in terms of the normal flux jumps at the flux points $[\hat{F}]_{f,l}$, and the coefficients $\alpha_{j,f,l}$ be determined, which are constant for any straight-sided triangles. The coefficients for the DG schemes are included in Appendix 1. In the case of *k* = 1, the coefficients for the first the solution point are {2.5 0.5 -1.5 -1.5 0.5 2.5}. Therefore, the formula for the correction is

$$\delta_{i,1} = \frac{1}{|V_i|}\left[(2.5[\tilde{F}]_{1,1} + 0.5[\tilde{F}]_{1,2})S_1 + (-1.5[\tilde{F}]_{2,1} - 1.5[\tilde{F}]_{2,2})S_2 + (0.5[\tilde{F}]_{3,1} + 2.5[\tilde{F}]_{3,2})S_3\right]. \tag{2.25}$$

Although all the flux points coincide with the solution points, as shown in Figure 2, it is necessary to distinguish flux points according to which face they are located on because each face has a different normal direction. In addition, the flux points on each face are numbered independently for easy identification and implementation.

In the SV method, the weighting function is 1 within a partition of the element, and 0 elsewhere, for example, as shown in Figure 3 in the case of *k* = 1. Repeating the same with all the partitions, we obtain again *m* equations for *m* unknowns $\delta_{i,j}$, which can be uniquely solved. The coefficients for several SV schemes are also included in Appendix 1. The coefficients for the first solution point of the second order SV method are {2 0.2 -0.7 -0.7 0.2 2}, corresponding to the following formula

$$\delta_{i,1} = \frac{1}{|V_i|}\left[(2[\tilde{F}]_{1,1} + 0.2[\tilde{F}]_{1,2})S_1 + (-0.7[\tilde{F}]_{2,1} - 0.7[\tilde{F}]_{2,2})S_2 + (0.2[\tilde{F}]_{3,1} + 2[\tilde{F}]_{3,2})S_3\right]. \tag{2.26}$$

In the SD method, the correction field is computed based on the direct differential of a reconstructed flux vector, i.e.,

$$\delta_i = \nabla \bullet [\hat{\vec{F}} - \vec{F}(Q_i^h)], \tag{2.27}$$

where $\hat{\vec{F}}$ is a reconstructed flux vector from the Riemann fluxes at the element interface and fluxes in the interior of the element. The components of $\hat{\vec{F}}$ are polynomials of degree *k+1*, one degree higher than the solution polynomial, so that δ_i is of degree *k*. The derivation is a little more involved than those for the DG and SV method. We found that only on an equilateral triangular grid can the SD method degenerate into the CPR form given in (2.16). This is not surprising because the SD method is generally not only dependent on the normal fluxes at element interfaces, but also on the tangential fluxes. The *k = 1* linear case has the following coefficients $\alpha_{j,f,l}$ at the first solution point {2 0 -0.5 -0.5 0 2}, resulting in the following formula

$$\delta_{i,1} = \frac{1}{|V_i|}\left[2[\tilde{F}]_{1,1}S_1 + (-0.5[\tilde{F}]_{2,1} - 0.5[\tilde{F}]_{2,2})S_2 + 2[\tilde{F}]_{3,2}S_3\right]. \tag{2.28}$$

Note that the coefficients are quite different for the DG, SV and SD methods. These schemes have been numerically confirmed to be equivalent to the DG, SV and SD methods for linear conservation laws.

Since the DG method is the most popular among the high-order methods for compressible flow, let's compare the CPR formulation with the DG method in terms of cost, for the case of k = 2.

Here are the main operations in the conventional DG methods with surface and volume quadratures:

- Reconstruct the state variables at 6 volume quadrature points and 9 surface quadrature points.
- Compute the analytical fluxes at the 6 volume quadrature points, and Riemann fluxes at the 9 surface quadrature points.
- Multiply the fluxes with the quadrature weights, and the weighting functions or their gradients to form the residual;
- Multiply the residual by the inverse of the mass matrix.

The main operations in the CPR method include:

- Compute the analytical fluxes at the 6 solution points, and 9 Riemann fluxes at surface flux points.
- Multiply the fluxes with the differential quadrature weights to form the divergence of the flux vector, and add the penalty terms to form the residual.

Due to the special choice of DOFs, the reconstruction cost is completely avoided, and the mass matrix is avoided in the CPR method. In addition, it is not necessary to store the weighting functions or their gradients. Therefore, the CPR formulation is more efficient in both memory and CPU time.

3.3 Conservation Constraints for Lifting Coefficients

It should not be a surprise that more choices of coefficients are possible, and may even be beneficial, for example, to allow the largest possible time step, or preserve certain range of frequency contents better than the standard methods. However, since we are dealing with conservation laws, the schemes should be conservative with any coefficients. In order to guarantee conservation, the integral conservation laws must be satisfies, i.e.,

$$\int_{V_i} \frac{\partial Q_i^h}{\partial t} dV + \int_{\partial V_i} \tilde{F}^n (Q_i^h, Q_{i+}^h, \vec{n}) dS = 0 \qquad (2.29)$$

which corresponds to (2.6) with W = 1. This means the CPR method is fundamentally conservative at the element level. Next let’s derive the conservative constraints which must be satisfied by the lifting coefficients. Assume the following quadrature is used for the volume integral

$$\int_{V_i} Q_i^h dV = |V_i| \sum_j v_j Q_{i,j}^h \ , \tag{2.30}$$

where v_j are the quadrature weights for the volume integral. Then the first term in (2.29) becomes

$$\int_{V_i} \frac{\partial Q^h}{\partial t} dV = |V_i| \sum_j v_j \frac{\partial Q_{i,j}^h}{\partial t} = -|V_i| \sum_j v_j \left[\nabla \bullet \vec{F}(Q_{i,j}^h) + \frac{1}{|V_i|} \sum_{f \in \partial V_i} \sum_l \alpha_{j,f,l} [\tilde{F}]_{f,l} S_f \right]. \tag{2.31}$$

Since the quadrature rule is exact for degree k (or less) polynomials, we have

$$|V_i| \sum_j v_j \nabla \bullet \vec{F}(Q_{i,j}^h) = \int_{V_i} \nabla \bullet \vec{F}(Q_i^h) dV = \int_{\partial V_i} F^n(Q_i^h) dS \ . \tag{2.32}$$

According to (2.29), we obtain

$$|V_i| \sum_j v_j \left[\frac{1}{|V_i|} \sum_{f \in \partial V_i} \sum_l \alpha_{j,f,l} [\tilde{F}]_{f,l} S_f \right] = \int_{\partial V_i} \left[\tilde{F}^n - F^n(Q^h) \right] dS = \sum_{f \in \partial V_i} \sum_l w_l [\tilde{F}]_{f,l} S_f . \tag{2.33}$$

The LHS of (2.33) can be simplified into

$$\sum_j v_j \left[\sum_{f \in \partial V_i} \sum_l \alpha_{j,f,l} [\tilde{F}]_{f,l} S_f \right] = \sum_{f \in \partial V_i} \sum_l \sum_j v_j \alpha_{j,f,l} [\tilde{F}]_{f,l} S_f \ . \tag{2.34}$$

Comparing (2.34) and (2.33), we obtain the following condition

$$w_l = \sum_j v_j \alpha_{j,f,l} \ . \tag{2.35}$$

This is also the necessary and sufficient condition for the CPR formulation to be conservative. In the linear case, $w_l = 1/2$ and $v_j = 1/3$, the conservation condition is then

$$\frac{3}{2} = \sum_{j=1}^{3} \alpha_{j,f,l} \ , \tag{2.36}$$

which is satisfied by the DG, SV and SD schemes.

3.4 Extension to Mixed Grids

It is obvious that (2.9) is valid for arbitrary types of elements besides triangles. For viscous flow problems, it is often advantageous to have quadrilateral cells or prismatic cell near solid walls to resolve viscous boundary layers. In the present study focusing on 2D problems, we address mixed grids including both triangular and quadrilateral meshes. As mentioned earlier, it is straightforward to extend Huynh's FR approach [4] to quadrilateral meshes. In order to simplify the implementation, we assume the polynomial degree *k* to be the same for both the triangular and quadrilateral meshes. Furthermore, the flux points along the element interfaces are required to match each other, as shown in Figure 4. In order to achieve an efficient implementation, all elements are transformed from the physical domain (x, y) into a standard square element $(\xi,\eta) \in$[-1,1]x[-1,1] as shown in Figure 5. The transformation can be written as

$$\begin{pmatrix} x \\ y \end{pmatrix} = \sum_{i=1}^{K} M_i(\xi,\eta) \begin{pmatrix} x_i \\ y_i \end{pmatrix}, \tag{2.37}$$

where K is the number of points used to define the physical element, (x_i, y_i) are the Cartesian coordinates of those points, and $M_i(\xi,\eta)$ are the shape functions. For the transformation given in (2.37), the Jacobian matrix J takes the following form

$$J = \frac{\partial(x, y)}{\partial(\xi,\eta)} = \begin{bmatrix} x_\xi & x_\eta \\ y_\xi & y_\eta \end{bmatrix}. \tag{2.38}$$

For a non-singular transformation, its inverse transformation must also exist, and the Jocobian matrices are related to each other according to

$$\frac{\partial(\xi,\eta)}{\partial(x, y)} = \begin{bmatrix} \xi_x & \xi_y \\ \eta_x & \eta_y \end{bmatrix} = J^{-1}.$$

Therefore the metrics can be computed according to

$$\xi_x = y_\eta / |J|, \xi_y = -x_\eta / |J|, \eta_x = -y_\xi / |J|, \eta_y = x_\xi / |J|. \tag{2.39}$$

The governing equations in the physical domain are then transformed into the computational domain (standard element), and the transformed equations take the following form

$$\frac{\partial \tilde{Q}}{\partial t} + \frac{\partial F^\xi}{\partial \xi} + \frac{\partial F^\eta}{\partial \eta} = 0. \tag{2.40}$$

where

$$\tilde{Q} = |J| \cdot Q \tag{2.41}$$

$$F^{\xi} = |J|(\xi_x F^x + \xi_y F^y) \tag{2.42}$$

$$F^{\eta} = |J|(\eta_x F^x + \eta_y F^y)\,. \tag{2.43}$$

Let $\vec{S}_{\xi} = |J|(\xi_x, \xi_y)$, $\vec{S}_{\eta} = |J|(\eta_x, \eta_y)$. Then we have $F^{\xi} = \vec{F} \bullet \vec{S}_{\xi}, F^{\eta} = \vec{F} \bullet \vec{S}_{\eta}$. In our implementation, $|J|$ and $\vec{S}_{\xi}$, $\vec{S}_{\eta}$ are stored at the solution points. Within the *ith* element, the solution polynomial is a tensor product of 1D Lagrange polynomials, i.e.,

$$Q_i^h(\xi,\eta) = \sum_{l=1}^{k+1}\sum_{j=1}^{k+1} Q_{i;j,l}^h L_j(\xi)\cdot L_l(\eta)\,, \tag{2.44}$$

where $Q_{i;j,l}^h$ are the state variables at the solution point (*j*,*l*), with *j* the index in ξ direction and *l* the index in η direction, $L_j(\xi)$ and $L_l(\eta)$ are 1D Lagrange polynomials in ξ and η directions. Based on the reconstructed solution $Q_i^h(\xi,\eta)$, the fluxes can be defined using $F^{\xi}(Q_i^h), F^{\eta}(Q_i^h)$. Again, one can also choose to represent the fluxes with Lagrange interpolation polynomials in the following form:

$$F_i^{\xi}(\xi,\eta) = \sum_{l=1}^{k+1}\sum_{j=1}^{k+1} F_{i;j,l}^{\xi} L_j(\xi)\cdot L_l(\eta)\,, \tag{2.45a}$$

$$F_i^{\eta}(\xi,\eta) = \sum_{l=1}^{k+1}\sum_{j=1}^{k+1} F_{i;j,l}^{\eta} L_j(\xi)\cdot L_l(\eta)\,. \tag{2.45b}$$

The reconstructed fluxes are only element-wise continuous, but discontinuous across cell interfaces. Again Riemann fluxes are computed at all four element interfaces in the normal directions, which are the same or opposite directions of $\vec{S}_{\xi}$ or $\vec{S}_{\eta}$. For example, at interfaces $\xi = -1$, and $\xi = 1$, the outgoing normals are

$$\vec{n}\big|_{\xi=-1} = -\vec{S}_{\xi} / \big|\vec{S}_{\xi}\big|\,. \tag{2.46a}$$

$$\vec{n}\big|_{\xi=1} = \vec{S}_{\xi} / \big|\vec{S}_{\xi}\big|\,. \tag{2.46b}$$

Therefore the Riemann flux corresponding to F^{ξ} is computed according to

$$\tilde{F}^{\xi}(-1,\eta) = -\tilde{F}^{n}(Q_i(-1,\eta), Q_{i+}(-1,\eta), \vec{n})\big|\vec{S}_{\xi}\big|\,, \tag{2.47a}$$

$$\tilde{F}^{\xi}(1,\eta) = \tilde{F}^{n}(Q_i(1,\eta), Q_{i+}(1,\eta), \vec{n})\big|\vec{S}_{\xi}\big|\,. \tag{2.47b}$$

Finally the DOFs are updated using the following equation

$$\begin{aligned}&\frac{\partial \tilde{Q}_{i;j,l}}{\partial t}+\frac{\partial F_i^{\xi}(\xi_{j,l},\eta_{j,l})}{\partial \xi}+\frac{\partial F_i^{\eta}(\xi_{j,l},\eta_{j,l})}{\partial \eta}\\ &+[\tilde{F}^{\xi}(-1,\eta_{j,l})-F_i^{\xi}(-1,\eta_{j,l})]g_L'(\xi_{j,l})+[\tilde{F}^{\xi}(1,\eta_{j,l})-F_i^{\xi}(1,\eta_{j,l})]g_R'(\xi_{j,l}) \qquad (2.48)\\ &+[\tilde{F}^{\eta}(\xi_{j,l},-1)-F_i^{\eta}(\xi_{j,l},-1)]g_L'(\eta_{j,l})+[\tilde{F}^{\eta}(\xi_{j,l},1)-F_i^{\eta}(\xi_{j,l},1)]g_R'(\eta_{j,l})=0.\end{aligned}$$

Note that, the correction is done in a "one dimensional" manner. For example, $[\tilde{F}]_{f,2}$ in Figure 4 only corrects the DOFs at the solid squares. In other words, for quadrilateral cells, the operations are actually one-dimensional, making the method more efficient per DOF than for triangular cells. The flux divergence in (2.48) can be obtained using either the LP or the CR approach. Numerical tests show that the CR approach gives more accurate and uniform results.

In the present implementation, we choose either the DG or the g_2 schemes from the FR family [4]. The g_2 scheme is the simplest possible. For all interior solution point, the correction vanishes. In the case shown in Figure 4, the correction for $[\tilde{F}]_{f,2}$ needs only to apply to the solution point coinciding with the flux location! Recall that the solution points for the g_2 approach are the Legendre-Lobatto points. Therefore, the solution points for all triangular elements at each edge are also Legendre-Lobatto points along the edge tangential direction to make the interface treatment as simple as possible.

3.5 Extension to Curved Boundary Cells

In second-order CFD solvers, curved boundaries are represented with linear line segments or planar facets. However, for high order solvers, this simple representation may not be able to preserve the geometry well enough since the computational grid is normally much coarser, and may even make the solvers unstable. The approach to solve the problem is to represent the curved boundary with higher order polynomials instead of the linear representation for second order solvers.

The change of boundary representations actually does not alter the solution technique, since all we need to do is to transform the curved boundary cell into a standard cell. For example, for a boundary element shown in Figure 6, a quadratic transformation is used to transform it to a standard element. After the transformation, the governing equation becomes (2.40) on the standard element, which has straight edges. The CPR formulation is then applied to the transformed equation on the standard right triangle. Eq. (2.17) for the transformed equation on a standard triangle becomes

$$\frac{\partial \tilde{Q}_{i,j}^{h}}{\partial t} + \nabla^{\xi} \bullet \vec{F}^{\xi}(\tilde{Q}_{i,j}^{h}) + 2\sum_{f\in\partial V_i}\sum_{l}\alpha_{j,f,l}[\tilde{F}^{\xi}]_{f,l}S_f^{\xi} = 0\,, \tag{2.47}$$

where superscript ξ mean the variables or operations evaluated on the computational domain. For example, $[\tilde{F}^{\xi}]_{f,l}$ are the normal jumps of the transformed fluxes across the faces of the standard triangle.

Note that when solving Equation (2.40), $\tilde{Q} = |J| \cdot Q$ is the state vector, and is assumed to be a degree k polynomial in the computational domain instead of Q. As a result, the derivative of Q should be calculated in the following way,

$$\frac{\partial Q}{\partial x} = \frac{1}{|J|}\left[\frac{\partial(|J|Q)}{\partial x} - \frac{\partial |J|}{\partial x}Q\right], \tag{2.48a}$$

$$\frac{\partial Q}{\partial y} = \frac{1}{|J|}\left[\frac{\partial(|J|Q)}{\partial y} - \frac{\partial |J|}{\partial y}Q\right]. \tag{2.48b}$$

Since the transformation is quadratic, $\frac{\partial |J|}{\partial x}$ and $\frac{\partial |J|}{\partial y}$ are non-zero. It is obvious that when the cell has straight faces, $|J|$ is a constant in the element. Then (2.48) reduces to the usual derivative evaluation.

For a quadrilateral cell, the curved boundary cells are treated in a similar manner. Each cell is transformed to a standard square with a high-order transformation, and then (2.40) is solved in the computational domain on a standard element.

3. Numerical Results

3.1 Accuracy Study with 2D Scalar Conservation Laws

Linear Wave Equation

In this case, we test the accuracy of the CPR method using the two-dimensional linear equation:

$$\frac{\partial u}{\partial t}+\frac{\partial u}{\partial x}+\frac{\partial u}{\partial y}=0, \qquad -1\le x\le 1, \quad -1\le y\le 1,$$
$$u(x,y,0)=u_0(x,y), \qquad \textit{periodic boundary condition}. \tag{3.1}$$

The initial condition is $u_0(x,y)=\sin\pi(x+y)$. Two types of computational meshes are used, one regular and the other irregular as shown in Figure 7. The finer meshes are generated recursively by cutting each coarser grid cell into four finer grid cells. The time integration schemes used are the TVD or SSP Runge-Kutta schemes of third or 4th order accuracy [5,2,6]. The results are made time step independent so that the spatial error is the dominant error source. The numerical simulation is carried until t = 1. The simulation was initialized by injecting the exact initial solution into the solution points. The numerical solutions at the three vertices of all the triangular elements are then used to compute the L_2 error norm to measure the global solution quality

$$Error_{L2}=\sqrt{\frac{\sum_{i=1}^{N}\sum_{j=1}^{3}(Q_{i,j}^{h}-Q_{i,j}^{e})^2}{3N}}, \tag{3.2}$$

where Q^e denotes the exact solution. For linear wave equations, all DG formulations are identical, so are all SV formulations. In the present CPR family, the DG and SV formulations are denoted by CPR-DG and CPR-SV respectively. Tables 1 and 2 present the computed errors with both formulations on the regular and irregular meshes. Note that optimal accuracy has been achieved for all the test cases, on both the regular and irregular meshes. The SD formulation cannot completely degenerate into the CPR, and therefore is not considered in the accuracy study.

2D Burgers Equation

In this case, we test the accuracy of the *CPR* method on the two-dimensional non-linear wave equation:

$$\frac{\partial u}{\partial t}+\frac{\partial u^2/2}{\partial x}+\frac{\partial u^2/2}{\partial y}=0, \qquad -1\le x\le 1, \quad -1\le y\le 1,$$
$$u(x,y,0)=\frac{1}{4}+\frac{1}{2}\sin\pi(x+y), \qquad \textit{periodic boundary condition}. \tag{3.3}$$

The initial solution is smooth. Due to the non-linearity of the Burgers equation, discontinuities will develop in the solution. At t = 0.1, the exact solution is still smooth. The numerical simulation is therefore carried out until t = 0.1 on both the regular and irregular grids. In Table 3-4, we present the L_1 , L_2 and L_∞ errors produced using the CPR-DG formulation on irregular grids, with both LP (Lagrange Polynomial, Table 3) approach and CR (chain-rule, Table 4) approach used in evaluating the interior flux derivatives and the accuracy of these two approaches is compared. Note in Table 3 and 4 that CR approach is better in every case and every norm of error considered here, since it not only produces smaller error, but also shows more consistency and better accuracy, especially for L_∞ norm, in grid refinement studies.

3.2 Accuracy Study with Vortex Evolution Problem

This is an idealized problem for the Euler equations in 2D used by Shu [3]. The mean flow is {*ρ*, *u*, *v*, *p*} = {1, 1, 1, 1}. An isotropic vortex is then added to the mean flow, i.e., with perturbations in *u*, *v*, and temperature *T* = *p*/*ρ*, and no perturbation in entropy $S = p/\rho^\gamma$:

$$
\begin{aligned}
(\delta u, \delta v) &= \frac{\varepsilon}{2\pi} e^{0.5(1-r^2)}(-y, x),\\
\delta T &= -\frac{(\gamma-1)\varepsilon^2}{8\gamma\pi^2} e^{1-r^2},\\
\delta S &= 0,
\end{aligned}
$$

where $r^2 = x^2 + y^2$, and the vortex strength *ε* = 5. If the computational domain is infinitely big, the exact solution of the Euler equations with the above initial conditions is just the passive convection of the isotropic vortex with the mean velocity (1, 1). In the numerical simulation, the computational domain is taken to be [-5,5]x[-5,5], with characteristic inflow and outflow boundary conditions imposed on the boundaries.

The numerical simulations were carried out until t = 2 on two different grids, one irregular triangular mesh and one mixed mesh as shown in Figure 7b and Figure 8. The finer irregular grids are generated recursively by cutting each coarser grid cell into four finer grid cells, while all mixed meshes are generated independently, which is mainly composed of irregular quadrilateral cell, with a small number of triangular cells. For irregular triangular mesh, two tests are performed. In Test 1, the interior flux derivatives are evaluated using the LP (Lagrange polynomial) approach, while in Test 2, these derivatives are evaluated by the CR (chain rule) approach described in Section 3.1. For the mixed mesh, only the latter approach is used. The CP-DG scheme is used for triangular cells while the FR-DG scheme is used for quadrilateral cells. For time integration, a 3-stage Runge-Kutta explicit scheme is used for time marching in all the cases.

In Table 5, the L_2 norms of density error at the solution points are presented for all three tests from 2nd order to 4th order. Comparing the results in Table 6, we find that the CR approach is much more accurate in every case. The CR approach not only produced the smaller errors, but also demonstrated more consistent order of accuracy in the grid refinement studies. In addition, the CPR method performs very well on the mixed grids, achieving the optimal order of accuracy on relatively poor quality meshes.

Because the CR approach in evaluating the interior flux derivatives is shown superior to the LP approach, it is used in all other test cases presented next.

3.3 Flow Over a Cylinder

To test the new method for curved wall boundaries, the case of a subsonic flow at Mach 0.3 around a half-cylinder is simulated. For the curve wall, a quadratic wall representation is used in the solver. Three grids are used, one triangular mesh, and two mixed mesh as shown in Figure 9(a), 10(a) and 11(a). The first mixed mesh is composed of regular quadrilateral cells near the wall boundary and irregular triangular cells elsewhere, while the second mixed mesh is composed of fully mixed irregular quadrilateral and triangular cells, and has poor grid quality near the wall.

The 2nd- 4th order CPR-DG scheme is used for triangular cells and the 2nd- 4th order FR-DG scheme used for the quadrilateral cells. For time integration, a LU-SGS implicit solver is used with 1st order time accuracy, and all the cases converged to machine zero.

The 2nd, 3rd and 4th order results for the 3 grids are shown in Figure 9, 10 and 11. Note that while 2nd order solutions show some discontinuities, smooth solutions are obtained with all the grids for 3rd and 4th order cases, even on Mixed Mesh #2, and all the 4th order solutions agree well with each other. The results show that the CPR method performs well for mixed mesh with curve boundary elements, since the smooth solutions across triangular and quadrilateral cells can be obtained as the order of accuracy increases. Furthermore, the high order solutions are also smooth near the curved boundary.

3.4 Flow Around NACA 0012 Airfoil

As the final test case for a more realistic geometry, the subsonic flow around a NACA 0012 airfoil is simulated with CPR method, with 2nd, 3rd and 4th order accuracy. The flow conditions are Mach = 0.3 and angle of attack $\alpha = 5$ degrees. Again curved boundaries are represented with the quadratic polynomials. The mesh, shown in Figure 12, is composed of regular quadrilateral mesh near the airfoil while irregular triangular and quadrilateral mesh elsewhere, with a total of 1,510 cells. The outer boundary of the computational domain is about 20 chords from the center of the airfoil. Again, the CPR-DG scheme is used for triangular cells and FR-DG is used for quadrilateral cells. A LU-SGS solver is used for time integration, and all the cases converge to machine zero.

The pressure and Mach number solutions from the 2nd, 3rd and 4th order simulation are shown in Figure 13-15. The computed solution for the 2nd order case is not smooth, with visible jump across cell boundaries, due to the coarse mesh, while in the 3rd and 4th order cases, the solution is quite smooth. Besides, the 3rd and 4th order results are almost the same with only a little difference near the trailing edge, which indicates a converged solution. In Figure 16, the entropy errors along the upper surface of the airfoil is plotted for the 2nd, 3rd and 4th order results as an indicator of the accuracy. Obviously the error shows an exponential convergence as the order of the simulation is increased.

4. Conclusions

In the present study, a remarkably simple and efficient flux reconstruction formulation developed by Huynh for conservation laws is extended and generalized to simplex cells. Instead of reconstructing the flux polynomial, the correction polynomial is reconstructed using a lifting operator. The new formulation is therefore called a *lifting collocation penalty* method. The final formulation is free of weighting functions, although the lifting coefficients are dependent on the weighting functions. Through a judicious selection of solution and flux points, solution reconstructions can be completely avoided. In addition, the mass matrix is always identity for arbitrary meshes. Furthermore, like the FR formulation, the discontinuous Galerkin, spectral volume, and in a special case the spectral difference method can all be recovered by the present formulation with different choices of lifting coefficients. The present formulation is more efficient than the original DG formulation based on volume and surface integral quadratures. For non-linear flux functions, the CPR method allows the interior flux derivatives to be computed “exactly” using the chain rule approach, which appears to be beneficial in solution accuracy. The extension to curved boundaries can be conducted in a straightforward manner because no surface or volume integrals are required. Numerical tests have been carried out for linear, nonlinear scalar conservation laws, and the 2D Euler equations, for mixed triangular and quadrilateral meshes. These tests fully demonstrate the capability of the CPR formulation. The extension to the Navier-Stokes equations, and to 3D mixed meshes including tetrahedral, prismatic, pyramidal, and hexahedral cells is currently carried out, and will be reported in future publications.

Reference

[1] Q. Chen and I. Babuska, Approximate optimal points for polynomial interpolation of real functions in an interval and in a triangle, Comput. Methods Appl. Mech. Engrg, 128, 405-417 (1995).

[2] Gottlieb S, Shu CW. Total variation diminishing Runge–Kutta schemes. Math Comput 1998;67:73–85.

[3] Hesthaven JS. From electrostatics to almost optimal nodal sets for polynomial interpolation in a simplex, SIAM J. Numer. Anal. Vol. 35 No. 2 (1998) 655-676.

[4] H.T. Huynh, A flux reconstruction approach to high-order schemes including discontinuous Galerkin methods, AIAA Paper 2007-4079.

[5] C.-W. Shu, Total-Variation-Diminishing time discretizations, SIAM Journal on Scientific and Statistical Computing 9, 1073-1084 (1988).

[6] Spiteri, R.J., and Ruuth, S.J., (2002). A new class of optimal high-order strong-stabilitypreserving time discretization methods. SIAM J. Numer. Anal. 40, 469–491.

[7] K Van den Abeele and C Lacor. An accuracy and stability study of the 2D spectral volume method, submitted to Journal of Computational Physics.

[8] K Van den Abeele, C. Lacor and Z.J. Wang, "On the connection between the spectral volume and the spectral difference method,” Journal of Computational Physics, Volume 227, No. 2, pp. 877-885 (2007).

[9] Z.J. Wang and Y. Liu, Spectral (finite) volume method for conservation laws on unstructured grids iii: one-dimensional systems and partition optimization, Journal of Scientific Computing, Vol. 20, No. 1, pp. 137-157 (2004).

Appendix

The Coefficients to Compute the Correction at Solution Points

- **2nd Order Schemes**

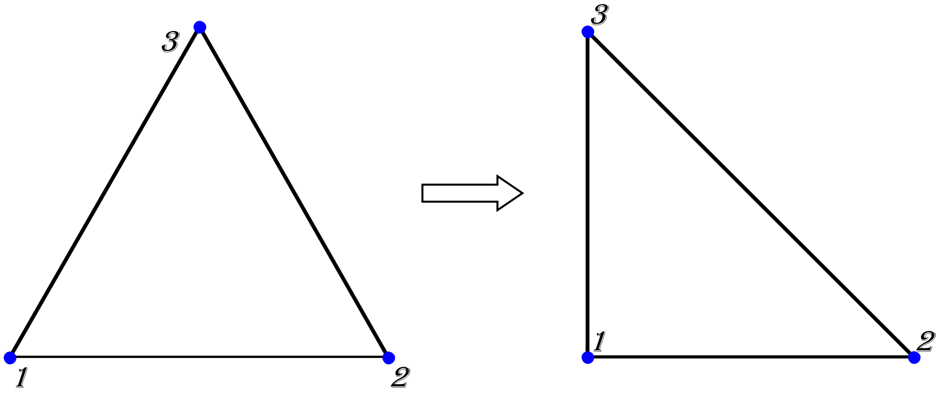


Figure A-1. Solution points for k = 1 element

Point Locations in Computational Domain

Point	ξ	η
1	0	0
2	1	0
3	0	1

$\alpha_{j,f,l}$ **for 2nd Order CPR-DG Scheme**

According to Equation (2.14), $\delta_{i,j} = \frac{1}{|V_i|} \sum_{f \in \partial V_i} \sum_{l} \alpha_{j,f,l} [\tilde{F}]_{f,l} S_f$. When applied to a arbitrary triangular cell in the physical domai, $|V_i|$ is the area of cell i , $[\tilde{F}]_{f,l}$ is the normal flux difference computed at l -th solution points along face/edge f , and S_f is simply the length of face/edge f . As a convention, faces are directional, and the direction of the faces is such that the interior of the cell is always on the left side of any face. Faces are numbered that within any cell, Face 1 are the face from Point 1 to Point 2, Face 2 from Point 2 to 3, and Face 3 from Point 3 to 1, and here Point 1, 2 and 3 are always the three vertices of the triangle. The numbering setup for 2nd order scheme is shown in Figure A-2.

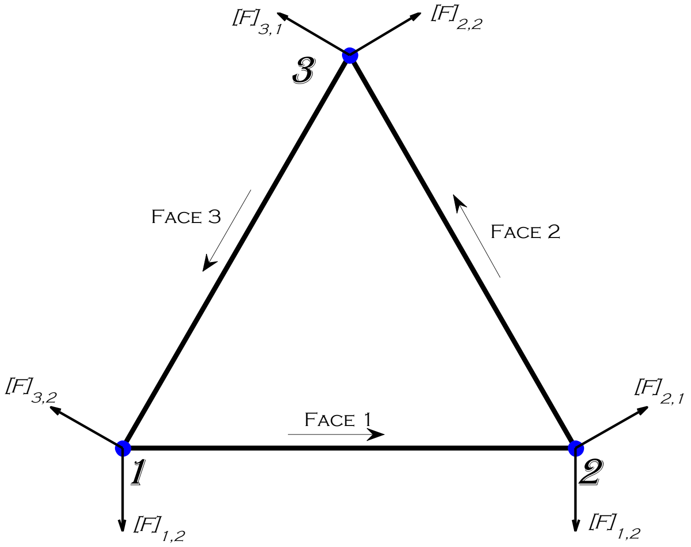


Figure A-2. Numbering Setup to Compute the Correction for k = 1 element

And $\alpha_{j,f,l}$ are listed in the following table:

j (f,l)	1
(1,1)	2.5
(1,2)	0.5
(2,1)	-1.5
(2,2)	-1.5
(3,1)	0.5
(3,2)	2.5

The table shows that

$$\delta_{i,1} = \frac{1}{V_i}(2.5[\tilde{F}]_{1,1}A_1 + 0.5[\tilde{F}]_{1,2}A_1 - 1.5[\tilde{F}]_{2,1}A_2 - 1.5[\tilde{F}]_{2,2}A_2$$
$$+ 0.5[\tilde{F}]_{3,1}A_3 + 2.5[\tilde{F}]_{3,2}A_3)$$

and from symmetry,

$$\delta_{i,2} = \frac{1}{V_i}(0.5[\tilde{F}]_{1,1}A_1 + 2.5[\tilde{F}]_{1,2}A_1 + 2.5[\tilde{F}]_{2,1}A_2 + 0.5[\tilde{F}]_{2,2}A_2$$
$$- 1.5[\tilde{F}]_{3,1}A_3 - 1.5[\tilde{F}]_{3,2}A_3)$$

$$\delta_{i,3} = \frac{1}{V_i}(-1.5[\tilde{F}]_{1,1} A_1 - 1.5[\tilde{F}]_{1,2} A_1 + 0.5[\tilde{F}]_{2,1} A_2 + 2.5[\tilde{F}]_{2,2} A_2$$
$$+ 2.5[\tilde{F}]_{3,1} A_3 + 0.5[\tilde{F}]_{3,2} A_3)$$

$\alpha_{j,f,l}$ **for 2nd Order CPR-SV Scheme**

j / (f,l)	1
(1,1)	2.0
(1,2)	0.2
(2,1)	-0.7
(2,2)	-0.7
(3,1)	0.2
(3,2)	2.0

$\alpha_{j,f,l}$ **for 2nd Order CPR-SD Scheme**

j / (f,l)	1
(1,1)	2.0
(1,2)	0.0
(2,1)	-0.5
(2,2)	-0.5
(3,1)	0.0
(3,2)	2.0

- **3rd Order Schemes**

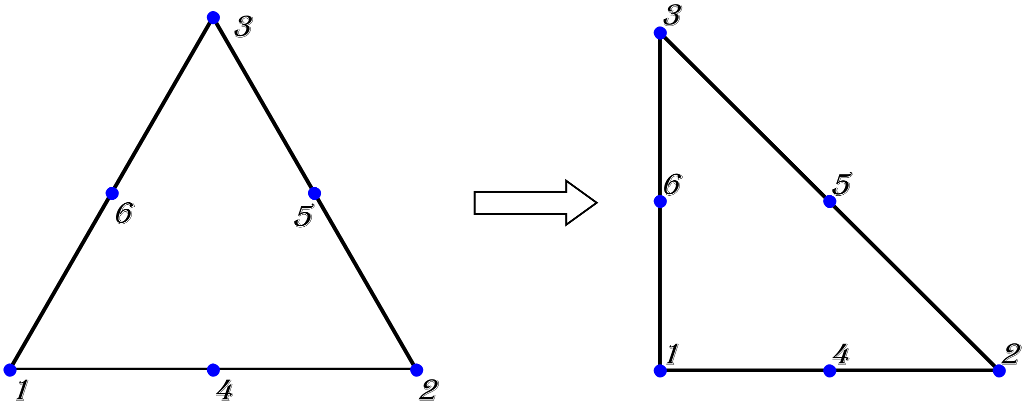


Figure A-3. Solution points for k = 2 element

Point Locations in Computational Domain

Point	ξ	η
1	0	0
2	1	0
3	0	1
4	0.5	0
5	0.5	0.5
6	0	0.5

$\alpha_{j,f,l}$ **for 3rd Order CPR-DG Scheme**

j (f,l)	1	4
(1,1)	4.5	0.5
(1,2)	2.0	5.0
(1,3)	-0.5	0.5
(2,1)	1.0	-0.625
(2,2)	4.0	-1.5
(2,3)	1.0	0.625
(3,1)	-0.5	0.625
(3,2)	2.0	-1.5
(3,3)	4.5	-0.625

$\alpha_{j,f,l}$ **for 3rd Order CPR-SV Scheme (with the partition "SV3P" by Van den Abeele[7])**

j (f,l)	1	4
(1,1)	6.53141002930328315	0.171052933692281709
(1,2)	0.739894614967283900	3.52565238715018730
(1,3)	-0.155939662412153337	0.171052933692281709
(2,1)	0.265607082915502158	-0.545239568903748461
(2,2)	0.369952521317455014	0.762826193575093650
(2,3)	0.265607082915502158	0.874186635211466751
(3,1)	-0.155939662412153337	0.874186635211466751
(3,2)	0.739894614967283900	-0.762826193575093650
(3,3)	6.53141002930328315	-0.545239568903748461

- **The 4th Order Schemes**

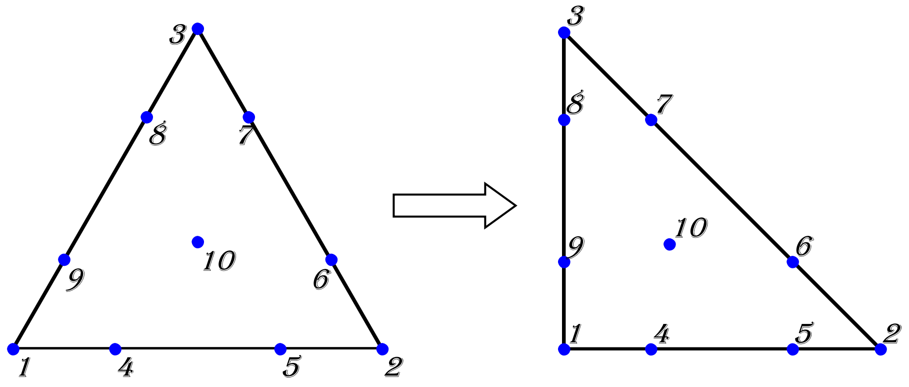


Figure A-4. Solution points for k = 3 element

Point Locations in the Computational Domain

Point	ξ	η
1	0	0
2	1	0
3	0	1
4	0.27639320225002103	0
5	0.72360679774997897	0
6	0.72360679774997897	0.27639320225002103
7	0.27639320225002103	0.72360679774997897
8	0	0.72360679774997897
9	0	0.27639320225002103
10	0.333333333333333333	0.333333333333333333

$\alpha_{j,f,l}$ for 4th Order CPR-DG Scheme

j (f,l)	1	4	10
(1,1)	7.	0.809016994374947424	0.0617283950617283951
(1,2)	4.04508497187473712	8.	-0.987654320987654321
(1,3)	-1.54508497187473712	1.5	-0.987654320987654321
(1,4)	0.5	-0.309016994374947424	0.0617283950617283951
(2,1)	-0.833333333333333333	0.259115050558307396	0.0617283950617283951
(2,2)	-4.16666666666666667	2.26295146066661059	-0.987654320987654321
(2,3)	-4.16666666666666667	0.0719684550415580098	-0.987654320987654321
(2,4)	-0.833333333333333333	-0.805180584266644234	0.0617283950617283951
(3,1)	0.5	-0.328152749066689099	0.0617283950617283951
(3,2)	-1.54508497187473712	1.07038187266672275	-0.987654320987654321
(3,3)	4.04508497187473712	-2.23863512170822468	-0.987654320987654321
(3,4)	7.	-0.292448383891640729	0.0617283950617283951

$\alpha_{j,f,l}$ for 4th Order CPR-SV Scheme (with the partition “SV4P” by Van den Abeele[7])

j (f,l)	1	4	10
(1,1)	11.203886398585652	0.10515345237391693	0.27526713373649432
(1,2)	2.65511402064370943	6.13135672232345356	-0.389968930474013208
(1,3)	-1.11620906021771755	0.647866338549474290	-0.389968930474013208
(1,4)	0.272083026510325471	-0.35670689389952049	0.27526713373649432
(2,1)	-0.235339114873132055	-0.225181007760767019	0.27526713373649432
(2,2)	-0.25648796757692409	0.473349519672136746	-0.389968930474013208
(2,3)	-0.25648796757692409	0.138909588123877733	-0.389968930474013208
(2,4)	-0.235339114873132055	-0.913139261303395950	0.27526713373649432
(3,1)	0.272083026510325471	-0.825187098917378380	0.27526713373649432
(3,2)	-1.11620906021771755	0.710864180742611977	-0.389968930474013208
(3,3)	2.65511402064370943	-1.25299752342169654	-0.389968930474013208
(3,4)	11.203886398585652	-0.519507774714493536	0.27526713373649432

Table 1. Test of CP-DG for $u_t + u_x + u_y = 0$, *with* $u_0(x, y) = \sin \pi(x + y)$, *at* $t = 1$, for triangular mesh.

Polynomial degree k	*Grid size*	Regular Mesh		Irregular Mesh	
		L_2 *error*	Order	L_2 *error*	Order
1	10x10x2	2.44e-2	-	4.45e-2	-
	20x20x2	5.89e-3	2.05	1.05e-2	2.08
	40x40x2	1.46e-3	2.01	2.57e-3	2.03
	80x80x2	3.64e-4	2.00	6.41e-4	2.00
2	10x10x2	1.88e-3	-	3.99e-3	-
	20x20x2	2.38e-4	2.98	5.14e-4	2.96
	40x40x2	2.98e-5	3.00	6.47e-5	2.99
	80x80x2	3.73e-6	3.00	8.10e-6	3.00
3	10x10x2	7.55e-5	-	2.59e-4	-
	20x20x2	4.94e-6	3.93	1.59e-5	4.03
	40x40x2	3.08e-7	4.00	9.91e-7	4.00
	80x80x2	1.93e-8	4.00	6.19e-8	4.00
5	10x10x2	7.53e-8	-	5.87e-7	-
	20x20x2	1.18e-9	6.00	9.22e-9	5.99
	40x40x2	1.85e-11	6.00	1.43e-10	6.01

Table 2. Test of CP-SV for $u_t + u_x + u_y = 0, with\ u_0(x, y) = \sin\pi(x + y), at\ t = 1$ for triangular mesh.

Polynomial degree k	*Grid size*	Regular Mesh		Irregular Mesh	
		L_2 *error*	Order	L_2 *error*	Order
1	10x10x2	5.94e-2	-	1.01e-1	-
	20x20x2	1.45e-2	2.03	2.62e-2	1.95
	40x40x2	3.72e-3	1.96	6.55e-3	2.00
	80x80x2	9.23e-4	2.01	1.63e-3	2.01
2	10x10x2	2.84e-3	-	7.47e-3	-
	20x20x2	3.71e-4	2.94	9.09e-4	3.04
	40x40x2	4.73e-5	2.97	1.13e-4	3.01
	80x80x2	5.97e-6	2.99	1.42e-5	2.99
3	10x10x2	1.04e-4	-	4.37e-4	-
	20x20x2	6.53e-6	3.99	2.58e-5	4.08
	40x40x2	4.11e-7	3.99	1.56e-6	4.05
	80x80x2	2.57e-8	4.00	9.61e-8	4.02

Table 3. Test of CP-DG for $u_t + uu_x + uu_y = 0,\ with\, u_0(x, y) = 0.25 + 0.5\sin\pi(x + y),\ at\ t = .1$, for triangular mesh with LP approach.

Polynomial degree k	*Grid size*	L_1 *error*	Order	L_2 *error*	Order	L_∞ *error*	Order
1	10x10x2	2.12e-2	-	2.65e-2	-	8.44e-2	-
	20x20x2	6.96e-3	1.61	9.96e-3	1.41	4.29e-2	0.98
	40x40x2	2.19e-3	1.67	3.75e-3	1.41	2.29e-2	0.91
	80x80x2	6.79e-4	1.69	1.38e-3	1.44	1.02e-2	1.17
2	10x10x2	4.36e-3	-	6.40e-3	-	3.01e-2	-
	20x20x2	7.76e-4	2.49	1.37e-3	2.20	9.91e-3	1.60
	40x40x2	1.26e-4	2.62	2.81e-4	2.29	3.01e-3	1.72
	80x80x2	1.93e-5	2.71	5.43e-5	2.37	8.17e-4	1.88
3	10x10x2	5.41e-4	-	9.59e-4	-	7.93e-3	-
	20x20x2	4.70e-5	3.52	1.05e-4	3.19	1.75e-3	2.18
	40x40x2	3.56e-6	3.72	9.86e-6	3.41	2.73e-4	2.68
	80x80x2	2.65e-7	3.75	8.48e-7	3.54	2.62e-5	3.38
5	10x10x2	1.15e-5	-	3.51e-5	-	6.20e-4	-
	20x20x2	2.94e-7	5.28	1.16e-6	4.92	3.26e-5	4.25
	40x40x2	6.37e-9	5.53	3.14e-8	5.21	1.57e-6	4.38
	80x80x2	1.23e-10	5.69	6.95-10	5.50	3.87e-8	5.34

Table 4. Test of CP-DG for $u_t + uu_x + uu_y = 0,\ with\, u_0(x,y) = 0.25 + 0.5\sin\pi(x+y),\ at\ t = .1$, for triangular mesh with CR approach.

Polynomial degree k	*Grid size*	L_1 *error*	Order	L_2 *error*	Order	L_∞ *error*	Order
1	10x10x2	1.34e-2	-	1.84e-2	-	7.33e-2	-
	20x20x2	3.64e-3	1.88	5.06e-3	1.86	2.57e-2	1.51
	40x40x2	9.39e-4	1.95	1.35e-3	1.91	8.63e-3	1.57
	80x80x2	2.37e-4	1.99	3.50e-4	1.95	2.60e-3	1.73
2	10x10x2	1.57e-3	-	2.75e-3	-	2.12e-2	-
	20x20x2	2.10e-4	2.90	4.04e-4	2.77	4.86e-3	2.13
	40x40x2	2.62e-5	3.00	5.50e-5	2.88	1.03e-3	2.24
	80x80x2	3.15e-6	3.06	7.27e-6	2.92	1.68e-4	2.62
3	10x10x2	1.31e-4	-	3.68e-4	-	4.13e-3	-
	20x20x2	8.64e-6	3.92	2.58e-5	3.83	4.50e-4	3.20
	40x40x2	5.70e-7	3.89	1.82e-6	3.83	3.62e-5	3.64
	80x80x2	3.63e-8	3.97	1.27e-7	3.84	2.77e-6	3.71
5	10x10x2	2.42e-6	-	1.07e-5	-	2.55e-4	-
	20x20x2	5.08e-8	5.57	2.61e-7	5.35	9.73e-6	4.71
	40x40x2	8.17e-10	5.96	4.45e-9	5.87	2.17e-7	5.49
	80x80x2	1.43e-11	5.84	7.99e-11	5.75	4.91e-9	5.47

Table 5. Test of CP-DG for Euler Equations with Vortex Propagation Case, for triangular and mixed meshes.

Polynomial degree k	*Grid size*	Irregular Triangular Mesh - Test 1		Irregular Triangular Mesh - Test 2		Mixed Mesh	
		L_2 *error*	Order	L_2 *error*	Order	L_2 *error*	Order
1	10x10x2	2.01e-2	-	1.39e-2	-	1.58e-2	-
	20x20x2	6.67e-3	1.59	4.41e-3	1.66	5.32e-3	1.57
	40x40x2	1.73e-3	1.95	1.08e-3	2.03	1.50e-3	1.83
	80x80x2	4.84e-4	1.84	2.54e-4	2.09	3.54e-4	2.08
2	10x10x2	7.14e-3	-	4.41e-3	-	2.95e-3	-
	20x20x2	1.07e-3	2.74	5.19e-4	3.09	5.62e-4	2.39
	40x40x2	1.60e-4	2.74	5.84e-5	3.15	7.42e-5	2.92
	80x80x2	2.29e-5	2.80	6.94e-6	3.07	8.63e-6	3.10
3	10x10x2	1.79e-3	-	6.70e-4	-	5.79e-4	-
	20x20x2	1.40e-4	3.68	4.79e-5	3.81	5.05e-5	3.52
	40x40x2	9.75e-6	3.84	2.96e-6	4.02	3.51e-6	3.85
	80x80x2	6.96e-7	3.81	1.71e-7	4.11	1.89e-7	4.22

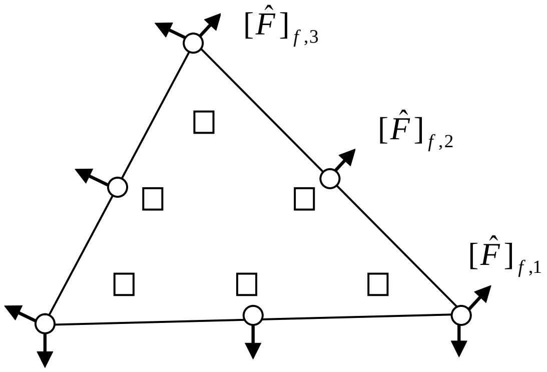
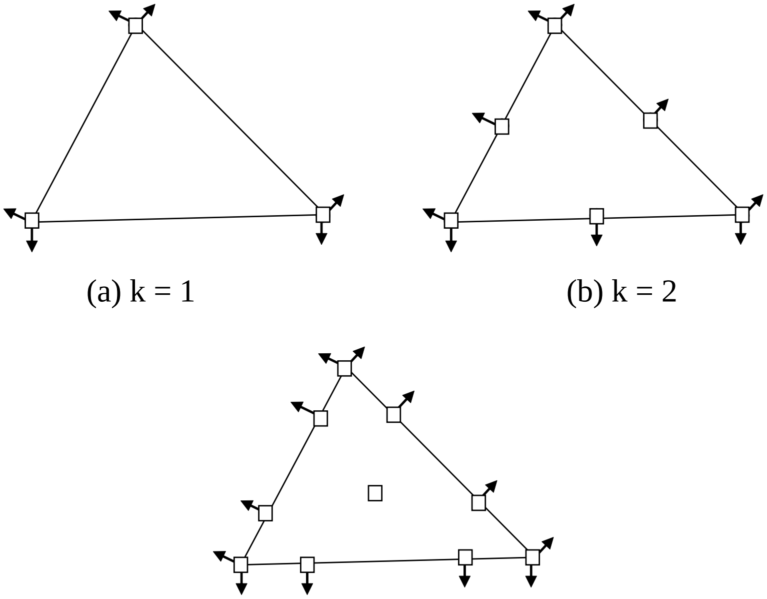


Figure 1. Solution points (squares) and flux points (circles) for k = 2.


(a) k = 1

(b) k = 2

(c) k = 3

Figure 2. Solution and flux points in a simplex for k = 1, 2, and 3.

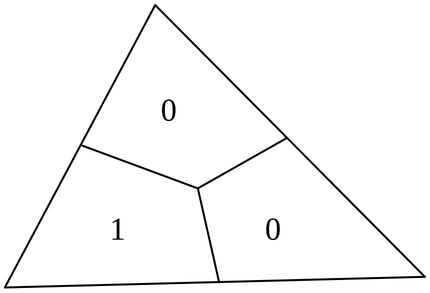


Figure 3. One of the weighting functions for the spectral volume method, k = 1.

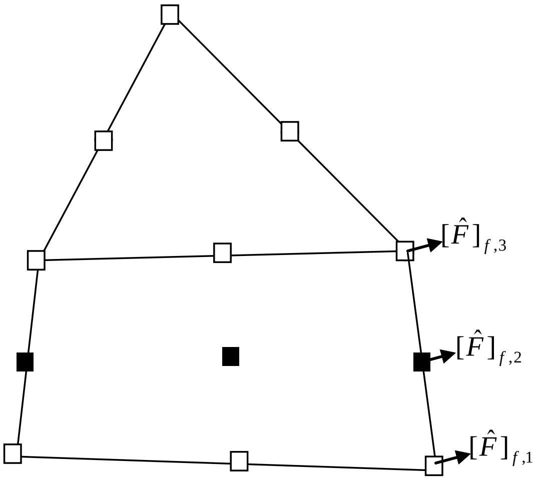


Figure 4. Solution and flux points for the 3rd order CP scheme on hybrid meshes.

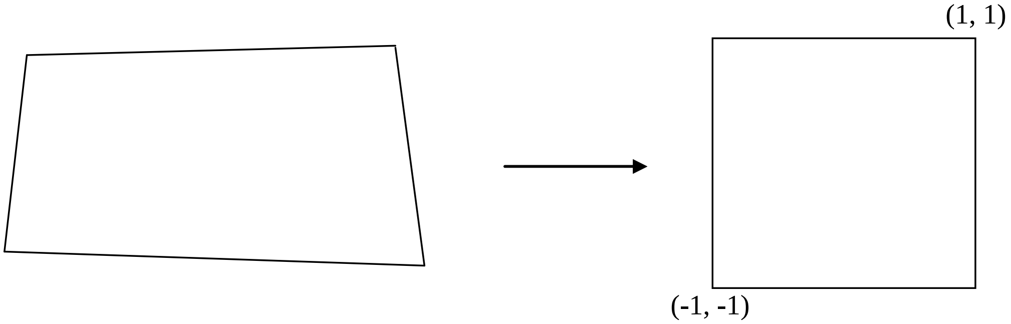


Figure 5. Transformation of a quadrilateral element to a standard element.

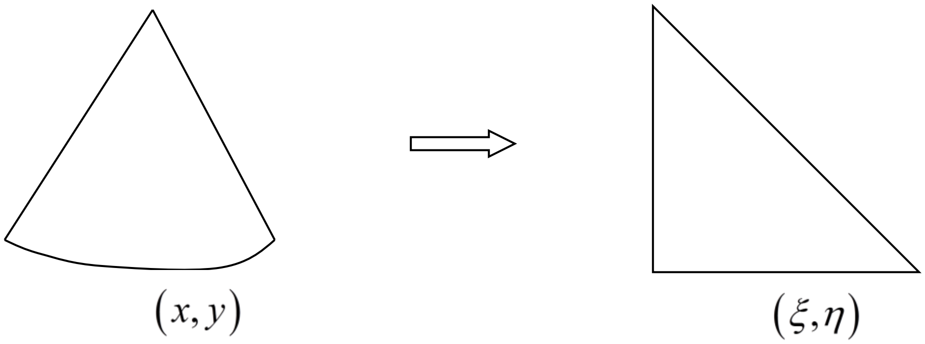
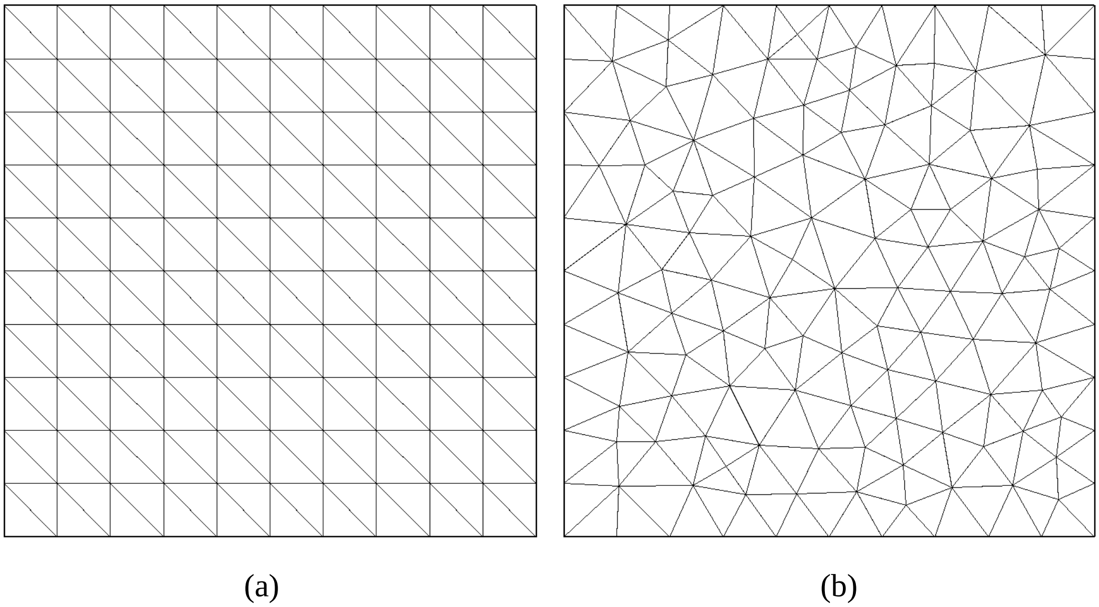


Figure 6. Transformation of a Curve Boundary Triangular Element to a Standard Element.



(a) (b)

Figure 7. Regular and Irregular "10x10x2" Computational Grids.

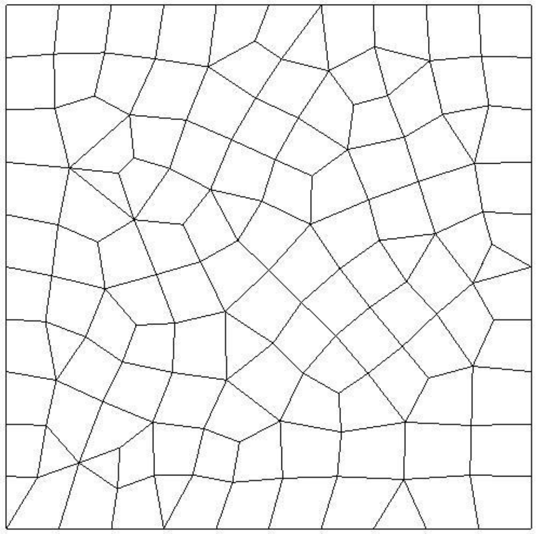
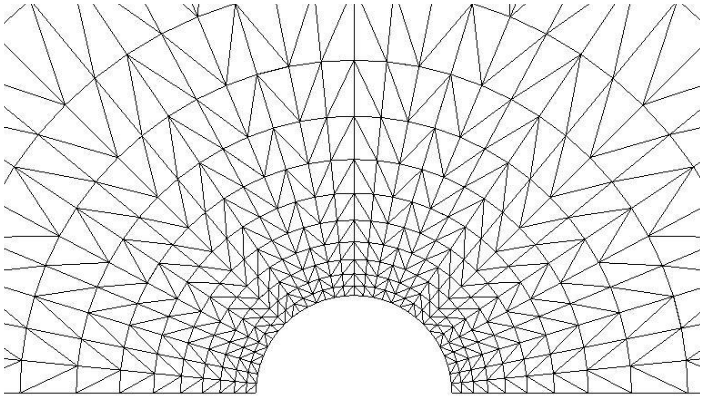
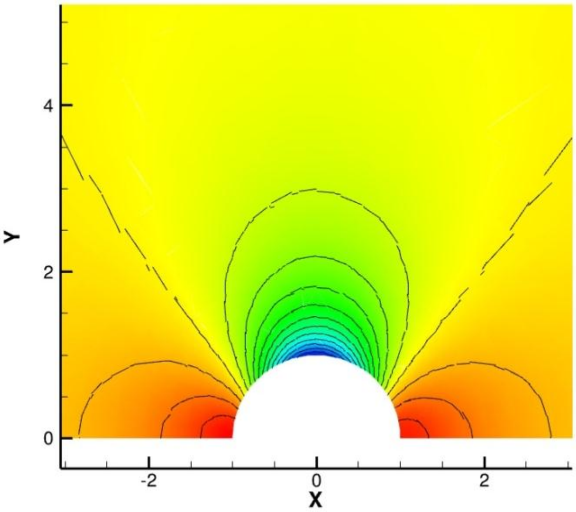


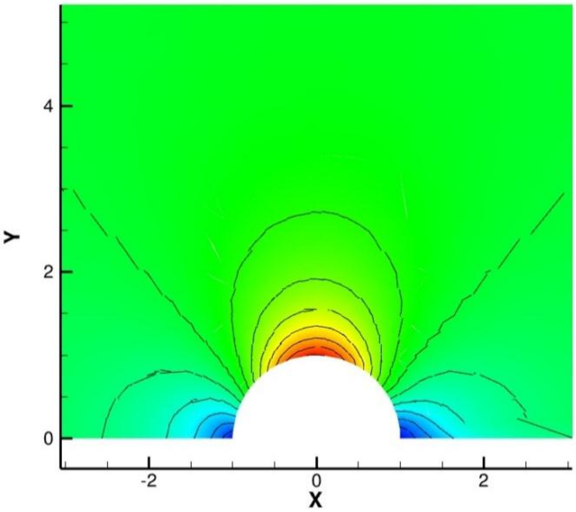
Figure 8. Coarse mixed mesh for accuracy study.



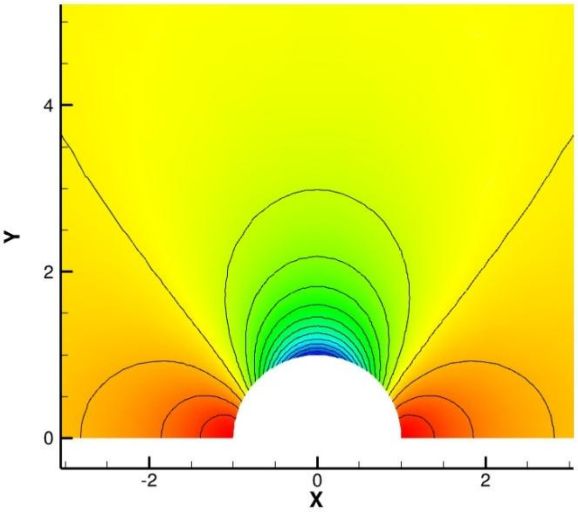
(a) Mesh



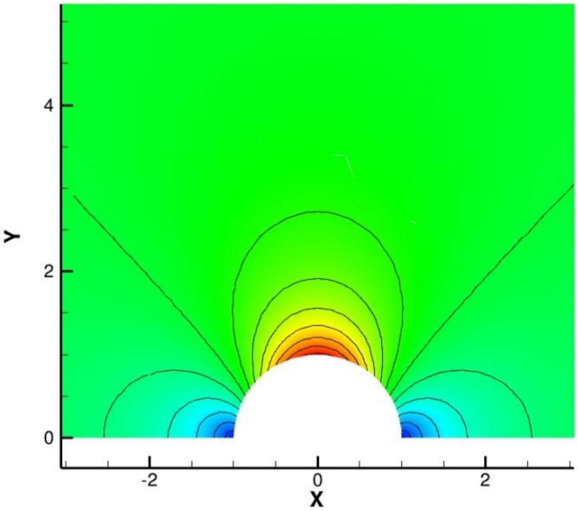
(b) Pressure Contour – 2nd order



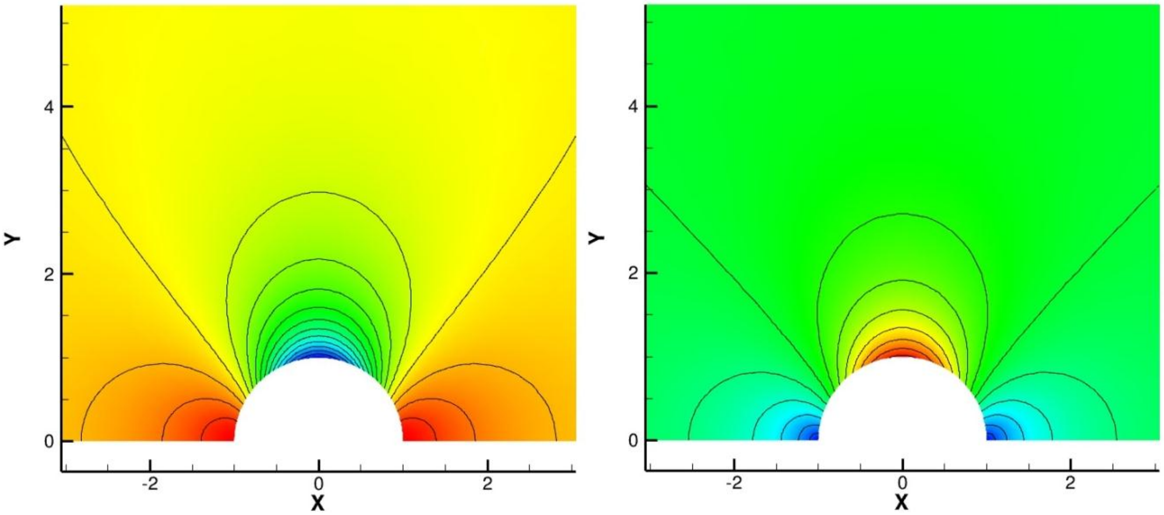
(c) Mach Number Contour—2nd order



(d) Pressure Contour – 3rd order

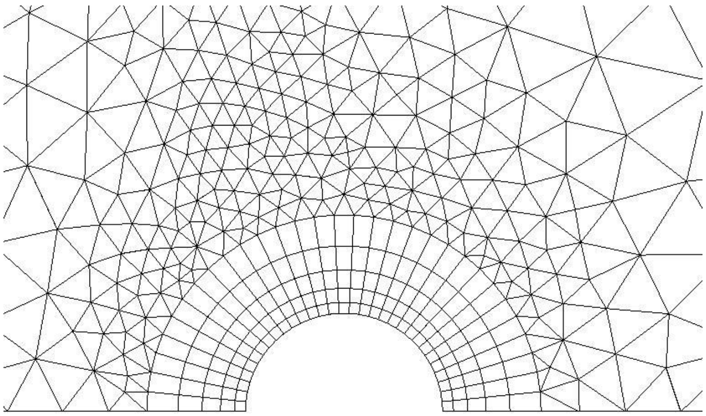


(e) Mach Number Contour—3rd order

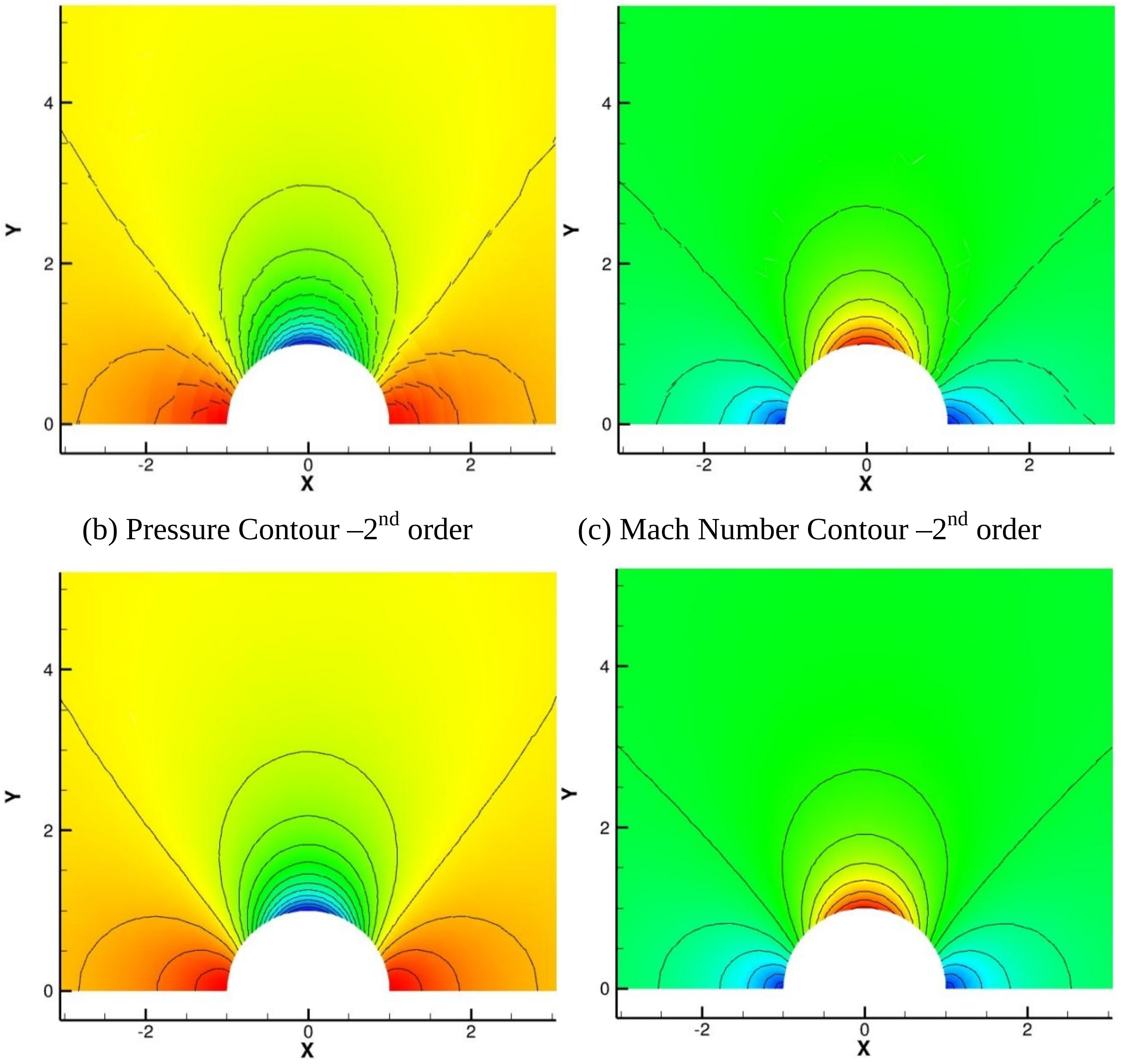


(f) Pressure Contour – 4th order (g) Mach Number Contour—4th order

Figure 9. Triangular Mesh and 2nd, 3rd and 4th Order Results for Flow around a Cylinder.

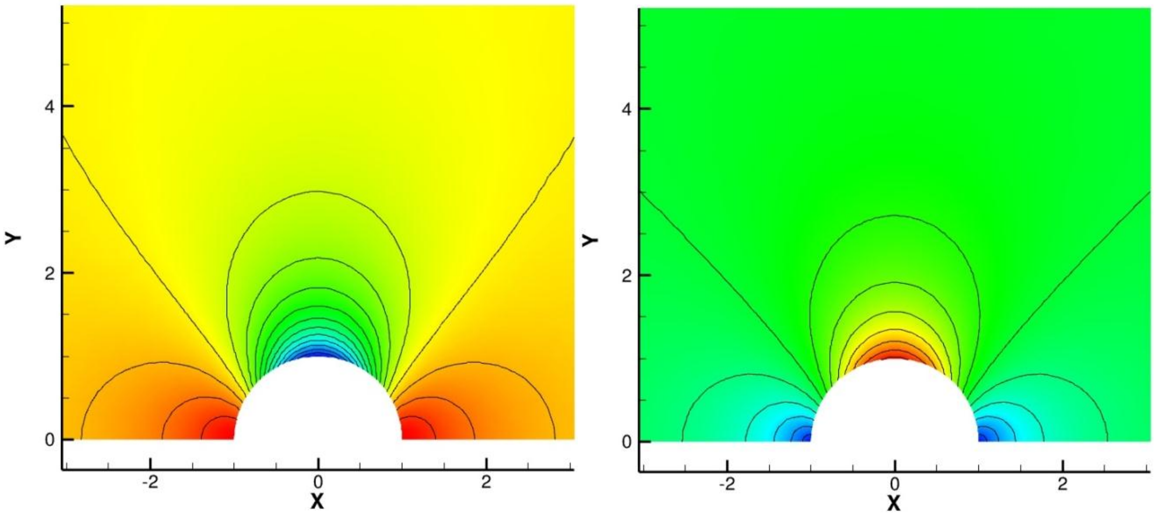


(a) Mesh



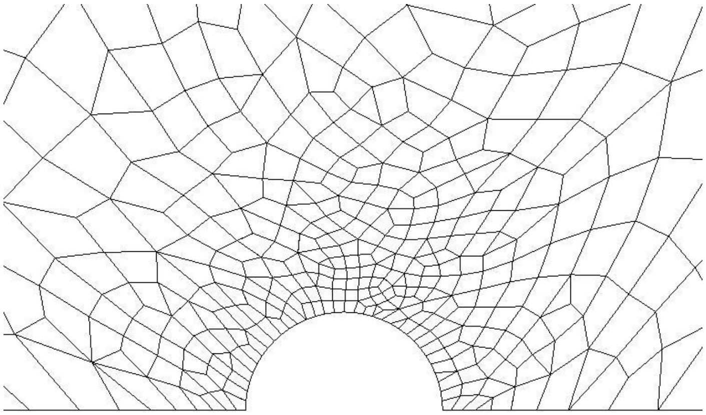
(b) Pressure Contour –2nd order (c) Mach Number Contour –2nd order

(d) Pressure Contour –3rd order (e) Mach Number Contour –3rd order

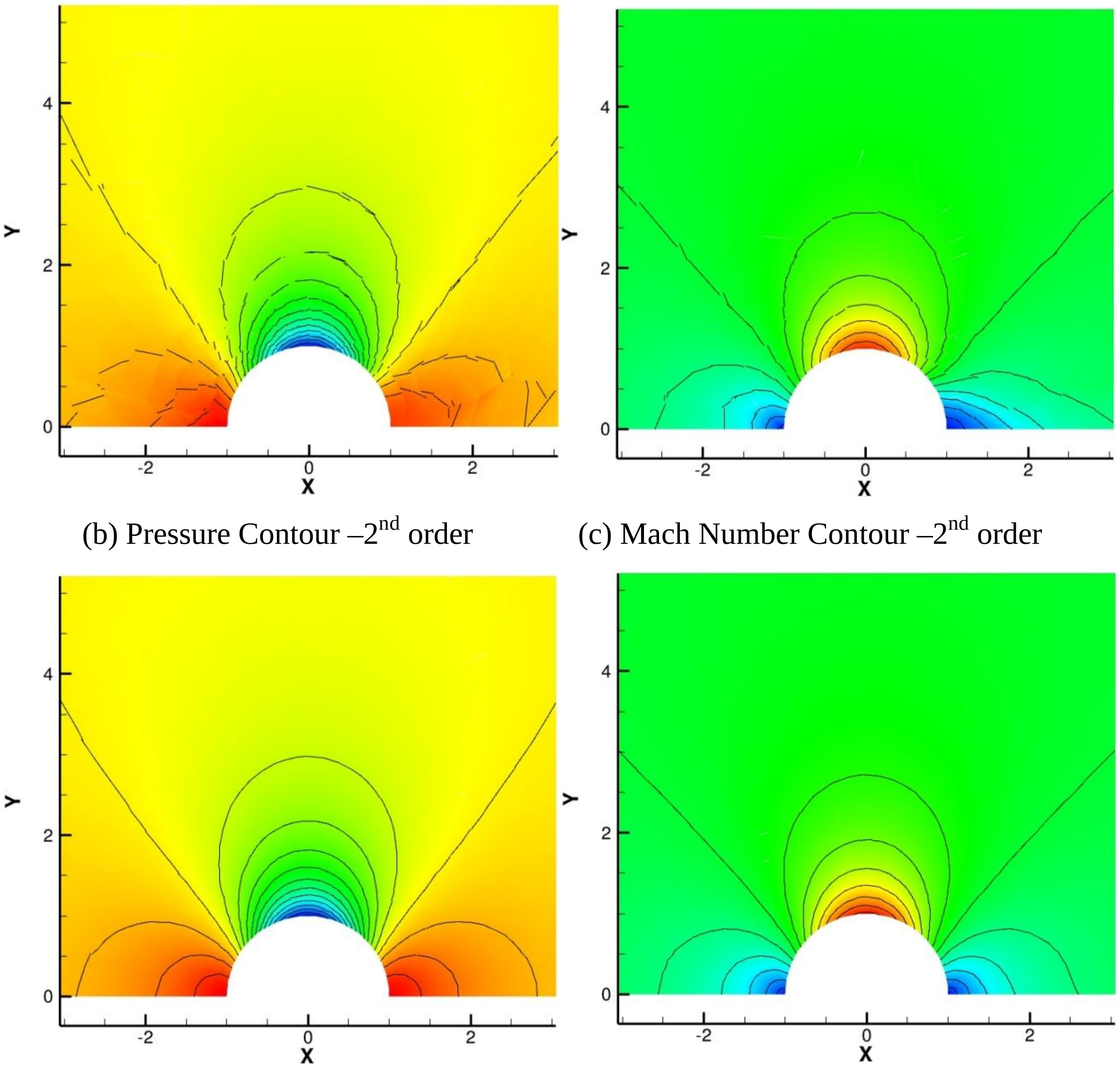


(f) Pressure Contour –4th order (g) Mach Number Contour –4th order

Figure 10. Mixed Mesh #1 and 2nd, 3rd and 4th Order Results for Flow around a Cylinder.

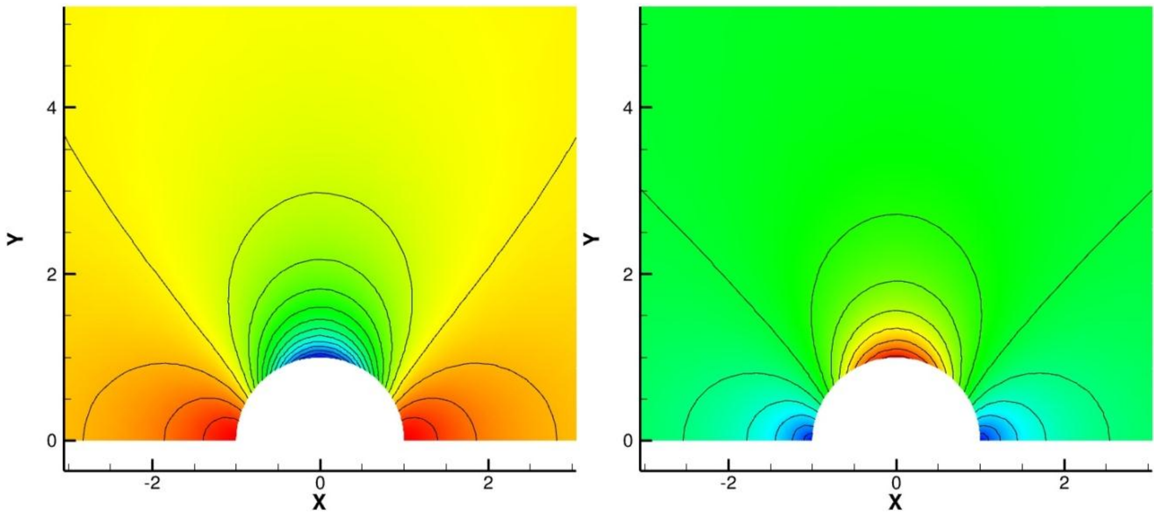


(a) Mesh



(b) Pressure Contour –2nd order (c) Mach Number Contour –2nd order

(d) Pressure Contour –3rd order (e) Mach Number Contour –3rd order



(d) Pressure Contour –3rd order (e) Mach Number Contour –3rd order

Figure 11. Mixed Mesh #2 and 4th Order Results for Flow around a Cylinder.

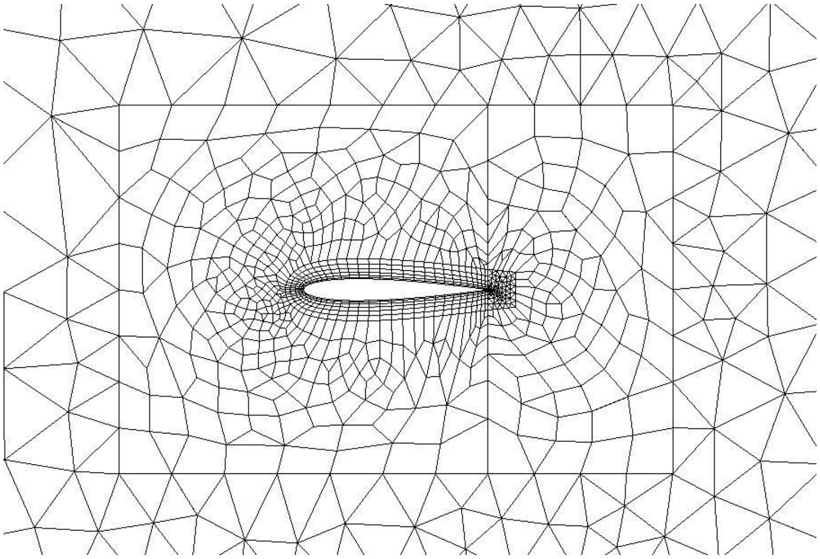
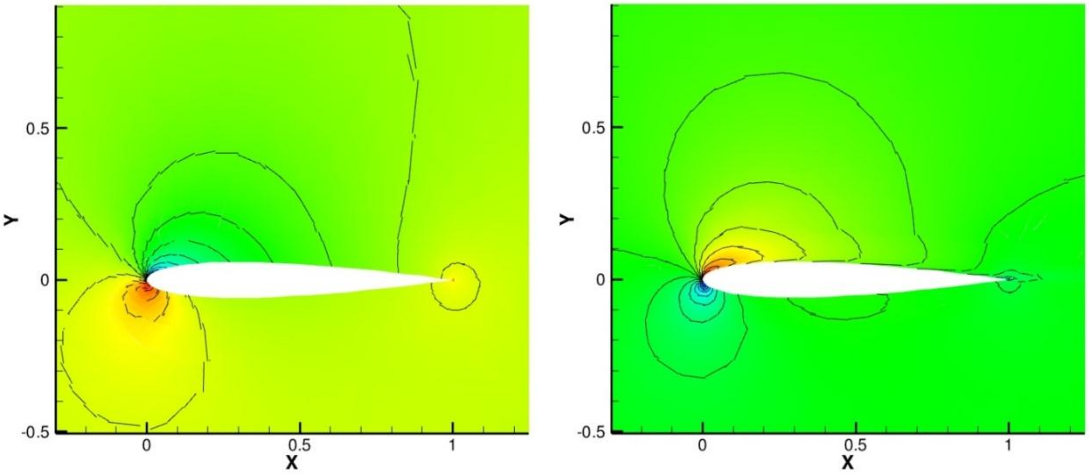
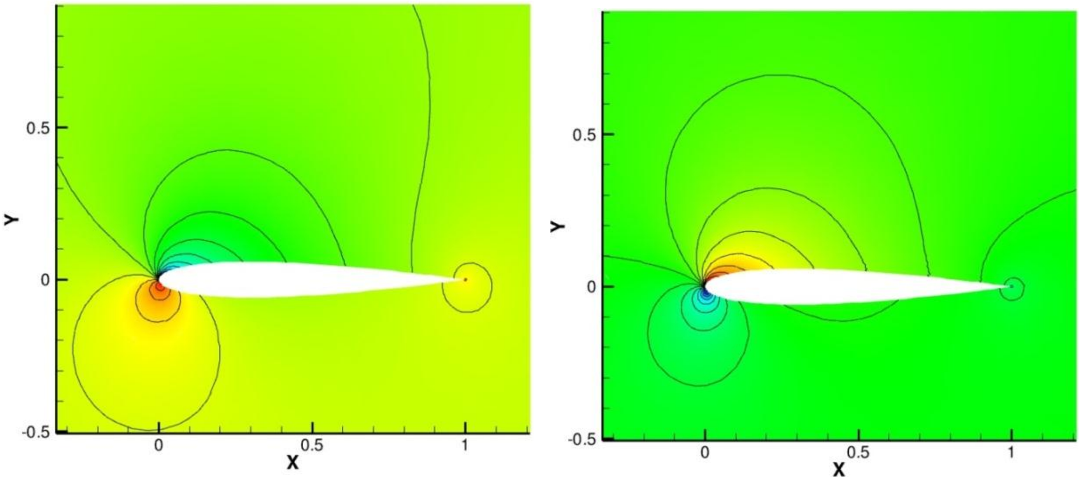


Figure 12. The Mixed Grid in the Simulation of Flow around NACA0012 Airfoil.



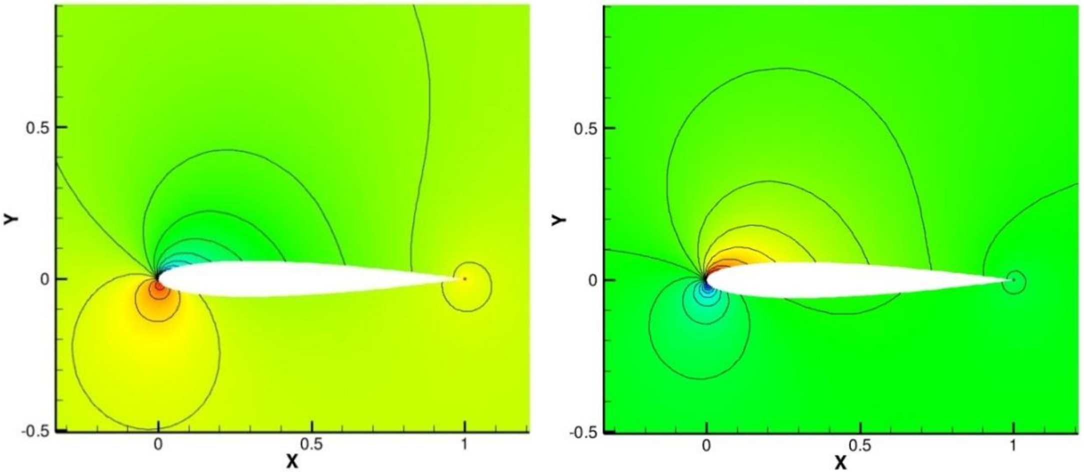
(a) Pressure Contour (b) Mach Number Contour

Figure 13. 2nd Order Solution of the Flow around NACA0012 Airfoil.



(a) Pressure Contour (b) Mach Number Contour

Figure 14. 3rd Order Solution of the Flow around NACA0012 Airfoil.



(a) Pressure Contour (b) Mach Number Contour

Figure 15. 4th Order Solution of the Flow around NACA0012 Airfoil.

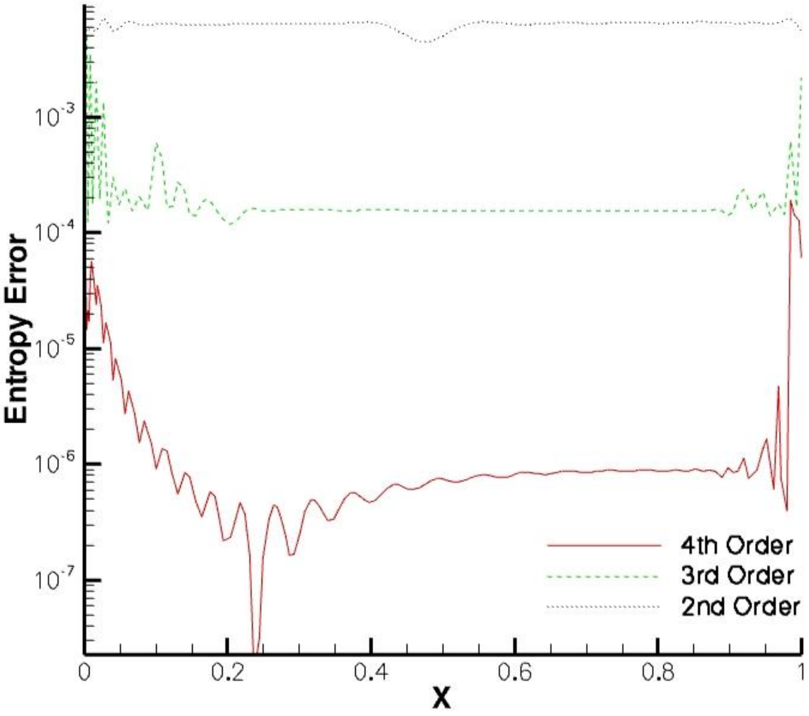


Figure 16. Entropy Error on the Upper Wall of NACA0012 Airfoil.

CHAPTER 3. Differential Formulation of Discontinuous Galerkin and Related Methods for the Navier-Stokes Equations

A paper submitted to the Communications in Computational Physics

Haiyang Gao, Z.J. Wang and H.T. Huynh

Abstract

A new approach to high-order accuracy for the numerical solution of conservation laws introduced by Huynh and extended to simplexes by Wang and Gao is renamed CPR (correction procedure or collocation penalty via reconstruction). The CPR approach employs the differential form of the equation and accounts for the jumps in flux values at the cell boundaries by a correction procedure. In addition to being simple and economical, it unifies several existing methods including discontinuous Galerkin, staggered grid, spectral volume, and spectral difference. To discretize the diffusion terms, we use the BR2 (Bassi and Rebay), interior penalty, compact DG (CDG), and I-continuous approaches. The first three of these approaches, originally derived using the integral formulation, were recast here in the CPR framework, whereas the I-continuous scheme, originally derived for a quadrilateral mesh, was extended to a triangular mesh. Fourier stability and accuracy analyses for these schemes on quadrilateral and triangular meshes are carried out. Finally, results for the Navier-Stokes equations are shown to compare the various schemes as well as to demonstrate the capability of the CPR approach.

1. Introduction

Second-order methods are currently popular in fluid flow simulations. For many important problems such as computational aeroacoustics, vortex-dominant flows, and large eddy and direct numerical simulation of turbulent flows, the number of grid points required by a second-order scheme is often beyond the capacity of current computers. For these problems, high-order methods hold the promise of accurate solutions with a manageable number of grid points. Numerous high-order methods have been developed in the last two decades. Here, we focus only on those that employ a polynomial to approximate the solution in each cell or element, and the polynomials collectively form a function which is discontinuous across cell boundaries. Commonly used methods of this type include discontinuous Galerkin (DG) [5-7,2,3], staggered-grid (SG) [15], spectral volume (SV) [24-28], and spectral difference (SD) [17-19]. Among these, DG and SV are usually formulated via the integral form of the equation, whereas SG and SD, the differential one. From an algorithm perspective, the difference among these methods lies in the definition of the degrees of freedom (DOFs), which determine the polynomial in each cell, and how these DOFs are updated.

High-order methods for conservation laws discussed above deal with the first derivative. Diffusion problems (viscous flows) involve the second derivative. There are many ways to extend a method of estimating the first derivative to the second; Arnold et al. analyzed several of them in [1]. Here, we restrict ourselves to approaches of compact stencil: the second derivative estimate in an element involves data of only that element and the immediate face neighbors. Such approaches have several advantages: the associated boundary conditions are simpler, the coding is easier, and the implicit systems are smaller. The four schemes of compact stencil employed are BR2 (Bassi and Rebay) [4], compact DG or CDG [20], interior penalty [9,12], and I-continuous (the value and derivative are continuous across the interface) [14]. The BR2 scheme, an improvement of the non-compact BR1 [2], is the first successful approach of this type for the Navier-Stokes equations. The CDG scheme is a modification of the local DG or LDG [8] to obtain compactness for an unstructured mesh. The interior penalty scheme is employed here with a penalty coefficient using correction function [14]. The I-continuous approach is highly accurate for linear problems on a quadrilateral mesh. Nicknamed “poor man’s recovery”, it can be considered as an approximation to the recovery approach of Van Leer and Nomura [23]. (The recovery approach is beyond the scope of this paper since, although it is more accurate than the schemes discussed here based on Fourier analysis [14], is more complex and costly.)

For conservation laws, Huynh (2007) [13] introduced an approach to high-order accuracy called flux reconstruction (FR). The approach solves the equations in differential form. It evaluates the first derivative of a discontinuous piecewise polynomial function by employing the straightforward derivative estimate together with a correction which accounts for the jumps at the interfaces. The FR framework unifies several existing methods: with appropriate correction terms, it recovers DG, SG, SV, SD methods. This framework was extended to diffusion problems using quadrilateral meshes in [14], where several existing schemes for diffusion were recast and analyzed. Wang and Gao (2009) [29] extended the FR idea to 2D triangular and mixed meshes with the lifting collocation penalty (LCP) formulation. The LCP methods was applied to solve the Euler and later Navier-Stokes equations in both two [10] and three dimensions [11]. Due to their tight connection, the FR and LCP methods are renamed *correction procedures via reconstruction* or CPR. The CPR formulation does not involve numerical integrations; the mass matrix inversion is built-in and therefore not needed; as a consequence, the approach is simpler and generally results in schemes more efficient than those by quadrature-based formulations.

In the present study, we demonstrate the capability of the CPR formulation for the numerical solutions of the Navier-Stokes equations as well as investigate the performance of several discretization approaches for diffusion. To accomplish these objectives and to prepare for complex geometries and viscous boundary layers, we first formulate the CPR framework on hybrid meshes of quadrilaterals and triangles. Next, we recast the four approaches for diffusion discussed above in this framework. Fourier stability and accuracy analyses of these schemes are then carried out on square and triangular meshes. Finally, results for several benchmark problems are shown.

The paper is organized as follows. The CPR formulation for hybrid meshes is presented in section 2. Section 3 describes the discretization of diffusion/viscous terms. Fourier analyses of the schemes for diffusion on square and triangular meshes are carried out in section 4. Numerical tests are shown in Section 5 for the Poisson and the Navier-Stokes equations. Conclusions are drawn in Section 6.

2. Governing Equations and Numerical Formulations

2.1 Governing Equations

For the scalar case, we consider the heat equation

$$u_t - \nabla^2 u = -f \tag{2.1}$$

with appropriate boundary conditions. Our goal is to obtain numerical solutions for the Navier-Stokes equations in conservative form

$$\frac{\partial Q}{\partial t}+\frac{\partial F}{\partial x}+\frac{\partial G}{\partial y}=0 \tag{2.2}$$

where Q is the vector of conserved variables, and F and G are the flux vectors formed by the invicid and viscous parts, $F=F^i-F^v, G=G^i-G^v$. More precisely,

$$Q=\left\{\begin{matrix}\rho\\ \rho u\\ \rho v\\ E\end{matrix}\right\}, F^i=\left\{\begin{matrix}\rho u\\ p+\rho u^2\\ \rho uv\\ u\left(E+p\right)\end{matrix}\right\}, G^i=\left\{\begin{matrix}\rho v\\ \rho uv\\ p+\rho v^2\\ v\left(E+p\right)\end{matrix}\right\}, \tag{2.3}$$

and

$$\begin{aligned}
F^v &= \mu\left\{\begin{matrix}0\\ 2u_x+\lambda\left(u_x+v_y\right)\\ v_x+u_y\\ u\left[2u_x+\lambda\left(u_x+v_y\right)\right]+v\left(v_x+u_y\right)+\frac{C_p}{\Pr}T_x\end{matrix}\right\},\\
G^v &= \mu\left\{\begin{matrix}0\\ v_x+u_y\\ 2v_y+\lambda\left(u_x+v_y\right)\\ u\left(v_x+u_y\right)+v\left[2v_y+\lambda\left(u_x+v_y\right)\right]+\frac{C_p}{\Pr}T_y\end{matrix}\right\}.
\end{aligned} \tag{2.4}$$

In (2.3)-(2.4), ρ is density, u and v velocity components in x and y directions, p pressure, E total energy, μ dynamic viscosity, C_p specific heat at constant pressure, Pr Prandtl number, and T temperature. For a perfect gas, pressure is related to total energy by

$$E=\frac{p}{\gamma-1}+\frac{1}{2}\rho\left(u^2+v^2\right). \tag{2.5}$$

The ratio of specific heats γ is assumed to be constant, and $\gamma=1.4$ for air. In addition, λ is set to -2/3 according to the Stokes hypothesis.

From (2.4), the viscous fluxes can be written as functions of both the conservative variables and their gradients,

$$F^v = F^v\left(Q,\vec{\nabla}Q\right), G^v = G^v\left(Q,\vec{\nabla}Q\right). \tag{2.6}$$

Note that the flux in the heat equation is a special case of the above: $F^v\left(Q,\vec{\nabla}Q\right)=Q_x$ and $G^v\left(Q,\vec{\nabla}Q\right)=Q_y$. Therefore, from here on, we employ only (2.6).

2.2 CPR (Correction Procedure via Reconstruction) Formulation

General CPR Formulation

The CPR formulation can be derived from a weighted residual method by transforming the integral formulation into a differential one. First, a hyperbolic conservation law can be written as

$$\frac{\partial Q}{\partial t}+\vec{\nabla}\bullet\vec{F}(Q)=0\,, \tag{2.7}$$

with proper initial and boundary conditions, where Q is the state vector and $\vec{F}=(F,G)$ is the flux vector. The computational domain is discretized into N non-overlapping elements $\{V_i\}$. Let W be an arbitrary weighting function. The weighted residual of Eq. (2.7) on element V_i can be written as

$$\int_{V_i}\left(\frac{\partial Q}{\partial t}+\vec{\nabla}\bullet\vec{F}(Q)\right)WdV=\int_{V_i}\frac{\partial Q}{\partial t}WdV+\int_{\partial V_i}W\vec{F}(Q)\bullet\vec{n}dS-\int_{V_i}\vec{\nabla}W\bullet\vec{F}(Q)dV=0. \tag{2.8}$$

Let Q_i be an approximate solution to Q at element i. We assume that in each element, the solution belongs to the space of polynomials of degree k or less, i.e., $Q_i\in P^k\left(V_i\right)$, (or P^k if there is no confusion) with no continuity requirement across element interfaces. The numerical solution Q_i should satisfy Eq. (2.8), i.e.,

$$\int_{V_i}\frac{\partial Q_i}{\partial t}WdV+\int_{\partial V_i}W\vec{F}(Q_i)\bullet\vec{n}dS-\int_{V_i}\vec{\nabla}W\bullet\vec{F}(Q_i)dV=0. \tag{2.9}$$

Since the solution is discontinuous across element interfaces, the above surface integral is not well-defined. To remedy this problem, a common normal flux is employed:

$$\vec{F}(Q_i)\bullet\vec{n} \approx F^n_{\text{com}}(Q_i\,,Q_{i+},\vec{n})\,, \tag{2.10}$$

where Q_{i+} is the solution on V_{i+}, which is outside V_i. Instead of (2.9), the solution is required to satisfy

$$\int_{V_i}\frac{\partial Q_i}{\partial t}W dV + \int_{\partial V_i} W F^n_{\text{com}}\,dS - \int_{V_i}\vec{\nabla} W\cdot\vec{F}(Q_i)dV = 0. \tag{2.11}$$

Applying integration by parts again to the last term of the above LHS, we obtain

$$\int_{V_i}\frac{\partial Q_i}{\partial t}W dV + \int_{V_i} W\vec{\nabla}\bullet\vec{F}(Q_i)dV + \int_{\partial V_i} W\left[F^n_{\text{com}} - F^n(Q_i)\right]dS = 0. \tag{2.12}$$

Here, we require test space to have the same dimension as the solution space. The test space is chosen in a manner to guarantee the existence and uniqueness of the numerical solution.

Note that the quantity $\vec{\nabla}\bullet\vec{F}(Q_i)$ involve no influence from the data in the neighboring cells; the influence of these data is represented by the above boundary integral, which is also called a "penalty term", penalizing the normal flux differences.

The next step is critical in the elimination of the test function. The boundary integral above is cast as a volume integral via the introduction of a "correction field" $\delta_i \in P^k$,

$$\int_{V_i} W\delta_i dV = \int_{\partial V_i} W[F^n]dS\,, \tag{2.13}$$

where $[F^n] = F^n_{\text{com}}(Q\ \,,Q^+,\vec{n}) - F^n(Q\ )$ is the normal flux difference. The above equation is sometimes referred to as the "lifting operator", which has the normal flux differences on the boundary as input and an element of $P^k\left(V_i\right)$ as output. Substituting Eq. (2.13) into Eq. (2.12), we obtain

$$\int_{V_i}\left[\frac{\partial Q_i}{\partial t}+\vec{\nabla}\bullet\vec{F}(Q_i)+\delta_i\right]WdV=0. \tag{2.14}$$

If the flux vector is a linear function of the state variable, then $\vec{\nabla}\bullet\vec{F}(Q_i)\in P^{k-1}$. In this case, the terms inside the square bracket are all elements of P^k. Because the test space is selected to ensure a unique solution, Eq. (2.14) is equivalent to

$$\frac{\partial Q_i}{\partial t}+\vec{\nabla}\bullet\vec{F}(Q_i)+\delta_i=0 \tag{2.15}$$

For non-linear conservation laws, $\vec{\nabla}\bullet\vec{F}(Q_i)$ is usually not an element of $P^k(V_i)$. As a result, we approximate it by its projection onto $P^k(V_i)$, denoted by $\Pi\left(\vec{\nabla}\bullet\vec{F}(Q_i)\right)$. One way to define this projection is

$$\int_{V_i}\Pi\left(\vec{\nabla}\bullet\vec{F}(Q_i)\right)W\mathrm{d}V=\int_{V_i}\vec{\nabla}\bullet\vec{F}(Q_i)W\mathrm{d}V \tag{2.16}$$

Eq. (2.14) then reduces to

$$\frac{\partial Q_i}{\partial t}+\Pi\left(\vec{\nabla}\bullet\vec{F}(Q_i)\right)+\delta_i=0. \tag{2.17}$$

With the introduction of the correction field δ_i, and a projection of $\vec{\nabla}\bullet\vec{F}(Q_i)$ for non-linear conservation laws, we have reduced the weighted residual formulation to a differential formulation, which involves no integrals.

Note that for δ_i defined by (2.13), if $W\in P^k$, Eq. (2.17) is equivalent to the DG formulation; if W varies on another space, the resulting δ_i is different, and we obtain a formulation corresponding to a different method [29].

Next, let the degrees-of-freedom (DOFs) be the solutions at a set of points $\{\vec{r}_{i,j}\}$ (*j* varies from 1 to *K*) called solution points (SPs), as shown in Figure 1. Then Eq. (2.17) holds true at the SPs, i.e.,

$$\frac{\partial Q_{i,j}}{\partial t}+\Pi\left(\vec{\nabla}\bullet\vec{F}(Q_{i,j})\right)+\delta_{i,j}=0, \tag{2.18}$$

where $\vec{\nabla}\bullet\vec{F}(Q_{i,j}) = \left[\vec{\nabla}\bullet\vec{F}(Q\ )\right]_{\vec{r}_{i,j}}$.

Along each edge, we also need to define *k*+1 points where the common normal fluxes are calculated; these points are called flux points (Figure 1). Once the solution points and flux points are chosen, the correction at the SPs can be written as

$$\delta_{i,j} = \frac{1}{|V_i|} \sum_{f\in\partial V_i} \sum_{l} \alpha_{j,f,l} [F^n]_{f,l} S_f , \tag{2.19}$$

where $\alpha_{j,f,l}$ are constants independent of the solution. Here, again, *j* is the index for solution points on each cell, *i* the cell index, *f* the face index, *l* the index of flux points on face *f*, and S_f the area.

Note that the correction for each solution point, namely, $\delta_{i,j}$ is a linear combination of all the normal flux differences on all the faces of the cell. Conversely, a normal flux difference at a flux point on a face, say (*f*, *l*) results in a correction at all solution points *j* of an amount $\alpha_{j,f,l}[F^n]_{f,l} S_f / |V_i|$.

Along each edge *f*, the flux difference values $[F^n]_{f,l}$ at the flux points defines a 1-D polynomial of degree *k* denoted by $\left[F^n\right]_f$. Eq. (2.13) can then be written as

$$\int_{V_i} W_k \sum_j L_j \delta_{i,j} dV = \sum_{f}^{f\in\partial V_i} \int_f W_k \left[F^n\right]_f dS. \tag{2.20}$$

Here L_j is the Lagrange polynomials, and $\underline{W_k}$ the test functions. This equation yields a linear system as *k* varies; the unknowns are $\delta_{i,j}$; both *k* and *j* vary from 1 to *K*. By setting $W_k = L_k$, the unknowns can be solved and the coefficients $\alpha_{j,f,l}$ can be determined. Note that after the approximation using the Lagrange polynomials (1D and 2D), the volume and surface integrals are carried out exactly in (2.20) for the solution of the linear system.

To compute the term $\Pi\left(\vec{\nabla}\bullet\vec{F}\left(Q_i\right)\right)$ efficiently, either chain-rule or Lagrange polynomial approach can be used, details about the two approaches can be found in Ref. [29].

Substituting (2.19) into (2.18) we obtain the following correction procedure via reconstruction (CPR) formulation

$$\frac{\partial Q_{i,j}}{\partial t} + \Pi\left(\vec{\nabla}\bullet\vec{F}(Q_{i,j})\right) + \frac{1}{|V_i|}\sum_{f\in\partial V_i}\sum_{l}\alpha_{j,f,l}[F^n]_{f,l}S_f = 0. \qquad (2.21)$$

It was shown that the location of SPs does not affect the numerical scheme for linear conservation laws [22, 13]. For efficiency, therefore, the solution points and flux points are always chosen to include corners of the cell; in addition, the solution points are chosen to coincide with the flux points along cell faces. Furthermore, in a computation with hybrid mesh, the flux points are always of the same distribution for different cell types for ease of interface treatment (Figure 2). For the 2D cases presented here, the Legendre Lobatto points along the edges are used as the flux points and also (part of) the solution points for both triangular and quadrilateral cells.

CPR for 1D

For the one-dimensional (1D) case, consider the element $V_i = [x_{i-1/2}, x_{i+1/2}]$; here, the flux $\vec{F}$ is F. Assuming a linear flux for derivation purpose, Eq. (2.21) takes the form

$$\frac{\partial Q_{i,j}}{\partial t} + \frac{\partial\left[F(Q_{i,j})\right]}{\partial x} + \alpha_{j,L}[F^n]_L + \alpha_{j,R}[F^n]_R = 0, \qquad (2.22)$$

where $[F^n]_L$ and $[F^n]_R$ are the flux differences on the left and right interfaces of the cell V_i. The quantities $\alpha_{j,L}$ and $\alpha_{j,R}$ are the correction coefficients. Recall that for the 1D case, the number of solution points is $K=k+1$; as a result, j varies from 1 to K, and $\alpha_{j,L}$ define a polynomial of degree $k=K-1$ as j varies. This polynomial is the derivative of a correction function, which approximates the zero function as explained below.

In 1D, the CPR method is equivalent to the flux reconstruction approach. The FR approach is a more general and flexible framework, making it very easy to obtain various sets of the coefficients $\alpha_{j,L}$. The FR approach for both convection and diffusion equations is documented by Huynh in [13] and [14].

Therefore, we now discuss (2.22) via the FR approach. Due to the discontinuity at the interfaces of the piecewise solution polynomial, the flux is also a piecewise discontinuous function. The idea of the FR method is to construct a new flux function $\hat{F}_i(x)$, which satisfies the following three criteria:

$\hat{F}_i(x)$ is a degree k+1 polynomial, i.e., one degree higher than the solution polynomial Q_i;

$\hat{F}_i(x)$ approximates $F_i\ (x)$ in some sense. In other words, some norm of the difference $\left\|\hat{F}_i(x)-F_i\ (x)\right\|$ is minimized;

At both ends of the element, the flux takes on the value of the common fluxes, i.e

$$\hat{F}_i(x_{i-1/2})=F_{\mathrm{com}}(Q_{i-1}(x_{i-1/2}),Q_i(x_{i-1/2}))\equiv F_{\mathrm{com},i-1/2}$$

$$\hat{F}_i(x_{i+1/2})=F_{\mathrm{com}}(Q_i(x_{i+1/2}),Q_{i+1}(x_{i+1/2}))\equiv F_{\mathrm{com},i+1/2}\,,$$

where, at each interface, $F_{\mathrm{com}}(Q^-,Q^+)$ is a common interface flux obtained from the two values Q^- and Q^+ to the left and right of that interface. For the inviscid flux, the Riemann or upwind flux is employed. For the diffusion or viscous flux, the common quantities will be discusses in section 4. Once this flux function $\hat{F}_i(x)$ is found, the DOFs are updated using the following differential equation

$$\frac{\partial Q_{i,j}}{\partial t}+\frac{\partial \hat{F}_i(x_{i,j})}{\partial x}=0\,. \tag{2.23}$$

Next, the reconstructed flux function is re-written as

$$\hat{F}_i(x)=F_i\ (x)+\sigma_i(x)\,, \tag{2.24}$$

where $\sigma_i(x)$ approximates the zero function. The function $\sigma_i(x)$ is further expressed as

$$\sigma_i(x)=[F_{\mathrm{com},i-1/2}-F_i\ (x_{i-1/2})]g_L(x)+[F_{\mathrm{com},i+1/2}-F_i\ (x_{i+1/2})]g_R(x)\,, \tag{2.25}$$

where $g_L(x)$ and $g_R(x)$ are called correction functions. They are of degree $k+1$, approximate the zero function, and satisfy

$$\begin{aligned} g_L(x_{i-1/2}) &= 1, \quad g_L(x_{i+1/2}) = 0 \\ g_R(x_{i-1/2}) &= 0, \quad g_R(x_{i+1/2}) = 1. \end{aligned} \tag{2.26}$$

For g_L, the condition $g_L(x_{i-1/2})$=1 deals with the jump at the left interface, and the condition $g_L(x_{i+1/2})$=0 leaves the right interface value unchanged. Eq. (2.22) then becomes

$$\frac{\partial Q_{i,j}}{\partial t} + \frac{\partial F_i\ (x_{i,j})}{\partial x} + [F_{\mathrm{com},i-1/2} - F_i\ (x_{i-1/2})]g_L'(x_{i,j}) + [F_{\mathrm{com},i+1/2} - F_i\ (x_{i+1/2})]g_R'(x_{i,j}) = 0\,. \tag{2.27}$$

Due to symmetry, we only need to consider $g_L(x)$, or simply $g(x)$. Using a linear transformation, it suffices to consider the correction function g on the standard element [−1, 1].Since g is a polynomial of degree k=K−1, it is defined by k+1 conditions. Two conditions are given by the first half of (2.26), namely $g(-1)=1$ and $g(1)=0$. Thus, k−1 conditions remain. The requirement of approximating the zero function can be satisfied by the condition that g is orthogonal to P^{k-2}, i.e.,

$$\int_{-1}^{1} g(\xi)\xi^k\, d\xi = 0,\ \ k = 0,1,\ldots,k-2 \tag{2.28}$$

Such a correction function is the Radau polynomial, and (2.22) results in a differential formulation of the DG method. For other correction functions, see [13, 14].

CPR for quadrilateral elements

The 1D CPR formulation can be extended to quadrilateral cells by tensor products. First, the mapping from the standard square [−1,1]x[−1,1] to a quadrilateral element (Figure 3) is given by

$$\begin{pmatrix} x \\ y \end{pmatrix} = \sum_{j=1}^{N_n} M_j(\xi,\eta) \begin{pmatrix} x_j \\ y_j \end{pmatrix}, \tag{2.29}$$

where x and y are the physical coordinates, ξ and η the local coordinates, (x_j, y_j) are the control points of the mapping (the four corners for straight-edge elements). N_n is the number of nodes used to define the physical element, $N_n = 4$ for elements with straight edges, $N_n > 4$ for elements with one or more curved edges. For each *j*, $M_j(\xi,\eta)$ is the shape function, which takes on the value 1 at node *j* and 0 at all other nodes. The Jacobian of the formulation is

$$J = \left|\frac{\partial(x,y)}{\partial(\xi,\eta)}\right| = \begin{vmatrix} x_\xi & x_\eta \\ y_\xi & y_\eta \end{vmatrix}. \tag{2.30}$$

Assuming that the transformation is nonsingular, the metrics can be computed by

$$\xi_x = y_\eta / J\,,\;\; \xi_y = -x_\eta / J\,,\;\; \eta_x = -y_\xi / J\,,\;\; \eta_y = x_\xi / J\,. \tag{2.31}$$

On the computational domain, the governing equation takes the form

$$\frac{\partial\tilde{Q}}{\partial t} + \frac{\partial\tilde{F}}{\partial\xi} + \frac{\partial\tilde{G}}{\partial\eta} = 0, \tag{2.32}$$

where

$$\tilde{Q} = J \cdot Q \tag{2.33}$$

$$\tilde{F} = J(\xi_x F + \xi_y G) \tag{2.34}$$

$$\tilde{G} = J(\eta_x F + \eta_y G)\,. \tag{2.35}$$

Next, set $\vec{S}_\xi = J(\xi_x,\xi_y)$, $\vec{S}_\eta = J(\eta_x,\eta_y)$. Then we have $\tilde{F} = \vec{F}\bullet\vec{S}_\xi, \tilde{G} = \vec{F}\bullet\vec{S}_\eta$. In our implementation, J and $\vec{S}_\xi$, $\vec{S}_\eta$ are stored at the solution points. Within the *ith* element, the solution polynomial is a tensor product of 1D Lagrange polynomials, i.e.,

$$Q_i\,(\xi,\eta) = \sum_{m=1}^{k+1}\sum_{j=1}^{k+1} Q_{i;j,m} L_j(\xi)\cdot L_m(\eta)\,. \tag{2.36}$$

There are K=(k+1)^2 solution points inside each element. Here $Q_{i;j,m}$ are the state variables at the solution point (j,m) of cell i, for clarity we use ";" to separate the cell index i from the two indices "j,m" of the solution point, with j the index in ξ direction, and m, η direction; $L_j(\xi)$ and $L_m(\eta)$ are 1D Lagrange polynomials in ξ and η directions, respectively. The fluxes $\tilde{F}_{i;j,m}$ and $\tilde{G}_{i;j,m}$ can be calculated by using $\tilde{F}(Q_{i;j,m}),\tilde{G}(Q_{i;j,m})$. The fluxes can be represented with Lagrange interpolation polynomials in the following form:

$$\tilde{F}_i\,(\xi,\eta)=\sum_{m=1}^{k+1}\sum_{j=1}^{k+1}\tilde{F}_{i;j,m}L_j(\xi)\cdot L_m(\eta)\,, \qquad (2.37a)$$

$$\tilde{G}_i\,(\xi,\eta)=\sum_{m=1}^{k+1}\sum_{j=1}^{k+1}\tilde{G}_{i;j,m}L_j(\xi)\cdot L_m(\eta)\,. \qquad (2.37b)$$

Common normal fluxes at the interfaces are computed at all four element edges as follows:

$$\tilde{F}_{\mathrm{com}}(-1,\eta)=-F^n_{\mathrm{com}}(Q_i(-1,\eta),Q_{i+}(-1,\eta),\vec{n})\left|\vec{S}_\xi\right|, \qquad (2.38a)$$

$$\tilde{F}_{\mathrm{com}}(1,\eta)=F^n_{\mathrm{com}}(Q_i(1,\eta),Q_{i+}(1,\eta),\vec{n})\left|\vec{S}_\xi\right|. \qquad (2.38b)$$

The equations for $\tilde{G}$ is similar. Finally the DOFs are updated using the following equation

$$\begin{aligned}&\frac{\partial\tilde{Q}_{i;j,m}}{\partial t}+\frac{\partial\tilde{F}_i\,(\xi_{j,m},\eta_{j,m})}{\partial\xi}+\frac{\partial\tilde{G}_i\,(\xi_{j,m},\eta_{j,m})}{\partial\eta}\\&+[\tilde{F}_{\mathrm{com}}(-1,\eta_{j,m})-\tilde{F}_i\,(-1,\eta_{j,m})]g_L'(\xi_{j,m})+[\tilde{F}_{\mathrm{com}}(1,\eta_{j,m})-\tilde{F}_i\,(1,\eta_{j,m})]g_R'(\xi_{j,m})\\&+[\tilde{G}_{\mathrm{com}}(\xi_{j,m},-1)-\tilde{G}_i\,(\xi_{j,m},-1)]g_L'(\eta_{j,m})+[\tilde{G}_{\mathrm{com}}(\xi_{j,m},1)-\tilde{G}_i\,(\xi_{j,m},1)]g_R'(\eta_{j,m})=0.\end{aligned} \qquad (2.39)$$

Note that the correction is done in a 1D manner.

CPR for triangular elements

The CPR method for triangular elements is computationally efficient: the coefficients $\alpha_{j,f,l}$ are not only independent of the solution, but also independent of the shape of the element. This means once the distribution of the solution points is chosen, the coefficients $\alpha_{j,f,l}$ can be computed analytically on a standard triangle by Eq. (2.20), and then applied to an arbitrary triangular element. The same comment holds for tetrahedral elements in 3D. The value of coefficients $\alpha_{j,f,l}$ for specific sets of solution points can be found in [29].

3. Compact schemes for discretization of the diffusion/viscous term

3.1 Basic Framework

We now use the CPR formulation discussed above to discretize the diffusion term. This discretization includes two parts: calculate the divergence of flux $\vec{\nabla}\bullet\vec{F}$ on the interior of the element and evaluate the common flux at the interface.

First, following [2], we introduce a new variable $\vec{R}$:

$$\vec{R} = \vec{\nabla} Q \tag{3.1}$$

Eq. (3.1) is solved using the weak formulation. In the CPR framework, the result is a collocation formulation:

$$\vec{R}_{i,j} = \left(\vec{\nabla} Q_i\right)_j + \frac{1}{|V_i|}\sum_{f\in\partial V_i}\sum_{l}\alpha_{j,f,l}[Q^{\text{com}} - Q_i]_{f,l}\vec{n}_f S_f\,, \tag{3.2}$$

where $Q_{f,l}^{\text{com}}$ is the common solution on interface *f*, and $Q_{i;f,l}$ is the solution within cell *i* on face *f*. The definition of $Q^{\text{com}} = Q^{\text{com}}\left(Q^{-}, Q^{+}, \vec{n}\right)$ varies with different approaches to discretize the diffusion terms. Here, with uniquely defined left and right side for each interface, Q^{-} and Q^{+} are the left and right solutions, and $\vec{n}$ the unit normal from left to right.

Next, the viscous fluxes at solution points are evaluated by

$$F_{i,j}^{v} = F^{v}\left(Q_{i,j}, \vec{R}_{i,j}\right), G_{i,j}^{v} = G^{v}\left(Q_{i,j}, \vec{R}_{i,j}\right). \tag{3.3}$$

Then, $\partial F^{v}/\partial x$ and $\partial G^{v}/\partial y$ can be obtained (using Lagrange polynomial). The divergence of the flux namely $\vec{\nabla}\bullet\vec{F}$ on the interior of the element can be calculated.

Besides those interior derivatives, common diffusion/viscous fluxes at the interfaces are also needed for the CPR formulation

$$F_{f,l}^{\text{com}} = F^{v}\left(Q_{f,l}^{\text{com}}, \vec{\nabla}Q_{f,l}^{\text{com}}\right), G_{f,l}^{\text{com}} = G^{v}\left(Q_{f,l}^{\text{com}}, \vec{\nabla}Q_{f,l}^{\text{com}}\right). \tag{3.4}$$

This means we also need to define a common gradient $\vec{\nabla}Q^{\text{com}} = \vec{\nabla}Q^{\text{com}}\left(\vec{\nabla}Q^{-}, \vec{\nabla}Q^{+}, Q^{-}, Q^{+}, \vec{n}\right)$ at the cell interfaces.

In the following subsections, we define Q^{com} and $\vec{\nabla}Q^{\text{com}}$ by the BR2, I-continuous, interior penalty and CDG schemes. Let the face *f* be fixed. We focus our attention on this face and the cells on both sides.

3.2 BR2

The common solution in BR2 is simply the average of solutions at two sides of the face

$$Q_{f,l}^{\text{com}} = \frac{Q_{f,l}^{-} + Q_{f,l}^{+}}{2}. \tag{3.5}$$

For the common gradient,

$$\vec{\nabla}Q_{f,l}^{\text{com}} = \frac{1}{2}\left(\vec{\nabla}Q_{f,l}^{-} + \vec{r}_{f,l}^{\,-} + \vec{\nabla}Q_{f,l}^{+} + \vec{r}_{f,l}^{\,+}\right). \tag{3.6}$$

where $\vec{\nabla}Q_{f,l}^{-}$ and $\vec{\nabla}Q_{f,l}^{+}$ are the gradients of the solution of the left and right cells with no correction, while $\vec{r}_{f,l}^{\,-}$ and $\vec{r}_{f,l}^{\,+}$ are the corrections to the gradients due to the common the solution on only the face *f*. More precisely,

$$\begin{aligned} \vec{r}_{f,l}^{\,-} &= \frac{1}{\left|V^{-}\right|}\sum_{m=1}^{N_{\text{fp}}} \beta_{l,m}^{-}[Q^{\text{com}} - Q^{-}]_{f,m}\vec{n}_{f}S_{f}, \\ \vec{r}_{f,l}^{\,+} &= \frac{1}{\left|V^{+}\right|}\sum_{m=1}^{N_{\text{fp}}} \beta_{l,m}^{+}[Q^{\text{com}} - Q^{+}]_{f,m}\left(-\vec{n}_{f}\right)S_{f} \end{aligned} \tag{3.7}$$

where N_{fp} is the number of solution points on face *f*, $\beta_{l,m}$ is the coefficient of correction due to face f. Note that the indices *l* and *m* vary on the face *f* and, for our choice of solution points, $\beta_{l,m} = \alpha_{j,f,m}$, where the index *j* for the solution points corresponds to the index *l* for the flux point on the face *f*. For triangular elements, $\beta_{l,m}$ are identical for any face *f* with a fixed distribution of flux points.

For quadrilateral elements, because the tensor product is used, $\vec{r}_{f,l}^{\,-}$ and $\vec{r}_{f,l}^{\,+}$ are thus computed in a 1D manner, which means the penalty matrix $\beta_{l,m}$ reduces to a diagonal matrix. Therefore, for a quadrilateral element, we only need to consider the coefficient $\gamma_l = \dfrac{\beta_{l,l} S_f}{V}$. Next, denote by $\tilde{\gamma}_l$ the quantity γ_l on the standard element [-1,1] X [1,1]. Then $\tilde{\gamma}_l = -g'(-1)$. This is also of the same form as the diffusion discretization of the FR method [14]. However, for a general quadrilateral element, $\tilde{\gamma}_l$ needs to be transformed back to γ_l in the physical domain (*x*, *y*) before being inserted to Eq. (3.7). Depending on the direction (*ξ* or *η*) we are dealing with, $\tilde{\gamma}_l$ and γ_l can be related by

$$\gamma_l = \sqrt{\xi_x^2 + \xi_y^2}\,\tilde{\gamma}_l = -\sqrt{\xi_x^2 + \xi_y^2}\,g'(-1) \text{ or}$$

$$\gamma_l = \sqrt{\eta_x^2 + \eta_y^2}\,\tilde{\gamma}_l = -\sqrt{\eta_x^2 + \eta_y^2}\,g'(-1). \qquad (3.8)$$

According to the Fourier analysis by Huynh [14], the derivative of Legendre polynomial $g_{LE}'(-1)$ offers the best accuracy for the linear diffusion equation, and therefore $g_{LE}'(-1)$ (the value is $-k(k+1)/2$ for degree *k* polynomial) is used in all the linear diffusion numerical tests presented later. For the N-S equations, however, a slightly larger coefficient $-(k+1)(k+2)/2$ is used to stabilize the scheme. The same coefficient is also used in the other 3 diffusion schemes.

3.3 I-continuous

The I-continuous approach in 1D was proposed by Huynh (2009) [14]. Its basic idea is: instead of prescribing a common solution Q^{com} at the interfaces, Q^{com} is an unknown to be solved by the condition that the corrected derivatives $\dfrac{\partial Q^C}{\partial x}$ is continuous across the interface *f*. Once we obtain Q^{com} , the common derivative $\frac{\partial Q^{\text{com}}}{\partial x}$ is also defined, since there is only one value at the interface.

In the 2D case, requiring the corrected gradient $\vec{\nabla} Q^{C}$ to be continuous at the interfaces would give us two conditions–continuity in *x* and *y* directions. Since we are only solving one variable Q^{com} , we require continuity only in the normal direction. The corrected gradients on the left (-) and right (+) can be expressed as

$$\begin{aligned} \vec{\nabla} Q^{C-}_{f,l} &= \vec{\nabla} Q^{-}_{f,l} + \frac{S_f^-}{|V^-|}\sum_{m=1}^{N_{\text{fp}}}[Q^{\text{com}} - Q^-]_{f,m}\beta^-_{l,m}\vec{n}, \\ \vec{\nabla} Q^{C+}_{f,l} &= \vec{\nabla} Q^{+}_{f,l} + \frac{S_f^+}{|V^+|}\sum_{m=1}^{N_{\text{fp}}}[Q^{\text{com}} - Q^+]_{f,m}\beta^+_{l,m}\left(-\vec{n}\right). \end{aligned} \tag{3.9}$$

Then we require the gradient to be continuous in the normal direction

$$\nabla Q^{C-}\cdot\vec{n} = \nabla Q^{C+}\cdot\vec{n}\,. \tag{3.10}$$

Substituting in (3.9), we have

$$\sum_{m=1}^{N_{\text{fp}}}\left(\frac{\beta^-_{l,m}S_f^-}{|V^-|}+\frac{\beta^+_{l,m}S_f^+}{|V^+|}\right)Q^{\text{com}}_{f,m} = \vec{\nabla} Q^{+}_{f,l}\cdot\vec{n} - \vec{\nabla} Q^{-}_{f,l}\cdot\vec{n} + \sum_{m=1}^{N_{\text{fp}}}\left(\frac{Q^-_{f,m}\beta^-_{l,m}S_f^-}{|V^-|}+\frac{Q^+_{f,m}\beta^+_{l,m}S_f^+}{|V^+|}\right) \tag{3.11}$$

Eq. (3.11) represents a linear system, from which $Q^{\text{com}}_{f,l}$ can be easily solved. Then, the common viscous flux $\vec{\nabla} Q^{\text{com}}$ is obtained by

$$\begin{gathered} \vec{\nabla} Q^{\text{com}}_{f,l}\cdot\vec{n} = \vec{\nabla} Q^{C-}\cdot\vec{n} \qquad \text{or} \qquad \vec{\nabla} Q^{\text{com}}_{f,l}\cdot\vec{n} = \vec{\nabla} Q^{C+}\cdot\vec{n} \\ \vec{\nabla} Q^{\text{com}}_{f,l}\cdot\vec{t} = \frac{\vec{\nabla} Q^{-}_{f,l}\cdot\vec{t} + \vec{\nabla} Q^{+}_{f,r}\cdot\vec{t}}{2}. \end{gathered} \tag{3.12}$$

Here $\vec{t}$ is the unit vector tangential to the face. The direction of $\vec{t}$ makes no difference provided that same $\vec{t}$ is used for both sides. In the case that one or both sides of the face is a quadrilateral cell, we can use $\gamma_l = \frac{\beta_{l,l}S_f}{V}$ in (3.11).

While the I-continuous approach is more involved than the other three schemes, it was shown in [14] via Fourier analysis that if g_{Le} is employed as the correction function, there is a significant gain in accuracy. Since such a Legendre-polynomial correction function is not yet known on triangles, we use the equivalence of the Radau polynomial.

Note that we need to solve a *k*+1 linear system for each face. The cost of this step is minimal since the matrices are independent of the solution. They only need to be inverted once during initialization. Therefore, the I-continuous approach is only slightly costlier than the BR2 approach.

3.4 Interior Penalty

Interior penalty is a simplified version of BR2 for triangular meshes, and is identical to BR2 for quadrilateral meshes. In BR2 the correction (or penalty) $\vec{r}^{\,+}_{f,l}$ and $\vec{r}^{\,-}_{f,l}$ at one face flux point is a linear combination of the solution differences of all points on the face. In interior penalty method, the penalty is only dependent on the solution difference at that point, i.e. the penalty is computed in a 1-D manner.

Therefore, equations (3.5) and (3.6) still holds for interior penalty method, the only difference lies in the computation of the corrections due to the common solution,

$$\vec{r}^{\,-}_{f,l} = -\left[Q^{\text{com}} - Q^{-}\right]_{f,l} g'(-1)\left(\frac{S_f}{V}\right)^{-} \vec{n}, \quad \vec{r}^{\,+}_{f,l} = \left[Q^{\text{com}} - Q^{+}\right]_{f,l} g'(-1)\left(\frac{S_f}{V}\right)^{+} \vec{n}, \tag{3.13}$$

$\frac{S_f}{V}\vec{n}$ here is in the place of terms like $\left(\xi_x, \xi_y\right)$ for the quadrilateral mesh, which represents the length scale factor.

3.5 CDG

The idea of CDG is to use one side of the face for the common solution and the other side for the common gradient.

For example, if we use the right (+) side for common solution and the left (-) side for common gradient, we obtain,

$$Q^{\text{com}}_{f,l} = Q^{+}_{f,l} \tag{3.14}$$

$$\vec{\nabla} Q^{\text{com}}_{f,l} = \vec{\nabla} Q^{-}_{f,l} + \vec{r}^{\,-}_{f,l}\,, \tag{3.15}$$

where

$$\vec{r}^{\,-}_{f,l} = \frac{1}{\left|V^{-}\right|} \sum_{m=1}^{N_{\text{fp}}} \beta^{-}_{l,m} \left[Q^{\text{com}} - Q^{-}\right]_{f,m} \vec{n}_f S_f \tag{3.16}$$

Alternatively, we can also use the opposite sides for common solution and common gradient. Note that the difference between the CDG and LDG methods is in the definition of the common gradient. At a fixed face, for CDG, the correction term $\vec{r}_{f,l}^{\,-}$ in 3.16 involves only the jumps at that face, whereas, for LDG, it involves the jumps at all the faces of the corresponding cell.

3.6 Boundary Conditions

In this subsection, we discuss the boundary conditions for the compact diffusion/viscous schemes. Let's assume a face *f* is on the boundary of the computational domain and the left side of *f* is the computational domain. Only the solutions and gradients on the left (-) side are available, and the final objective is to obtain Q^{com} and $\vec{\nabla} Q^{\mathrm{com}}$ for face *f*.

In the linear diffusion cases presented in Section 5, two types of boundary conditions are used. For Dirichlet boundary condition, Q^{+} is set to the boundary value while the gradient is extrapolated, i.e. $\nabla Q^{+} = \nabla Q^{-}$. Then the common quantities can be computed according to Section 3.2-3.5. For Neumann boundary conditions, the common solution is extrapolated $Q^{\mathrm{com}} = Q^{-}$, while ∇Q^{com} is set to the boundary value.

In the case of Navier-Stokes equations, the boundary condition is similar to the Dirichlet boundary condition for the linear case. Q^{+} can be simply fixed to the exact boundary values, or obtained through characteristic boundary conditions. For solid non-slipping walls, Q^{+} is set to Q^{-} with reversed velocity, resulting zero velocity for Q^{com}. The gradients are always extrapolated from the interior domain, i.e. $\nabla Q^{+} = \nabla Q^{-}$. For adiabatic walls, the normal temperature gradients are set to *0* when computing the fluxes by Equation (3.4).

4. Two-Dimensional Fourier (Von Neumann) Analysis

We now carry out the Fourier (Von Neumann) stability and accuracy analysis of the schemes discussed above on a square and a triangular mesh. On the domain $(-\infty,\infty)\times(-\infty,\infty)$, consider the diffusion equation

$$u_t = u_{xx} + u_{yy} \tag{4.1}$$

With initial condition $u_{\text{init}}(x) = e^{I(w_x x + w_y y)}$, where and $I = \sqrt{-1}$ (we use i for the cell index), and the wave numbers w_x and w_y lie between $-\pi$ and π. Low frequency data corresponds to wave numbers of small magnitude; high frequency, to those near $\pm\pi$. The exact solution is $u_{\text{exact}}(x) = e^{-(w_x{}^2 + w_y{}^2)t} e^{I(w_x x + w_y y)}$. At $(x, y, t) = (0, 0, 0)$,

$$(u_{\text{exact}})_t(0,0,0) = -(w_x{}^2 + w_y{}^2). \tag{4.2}$$

Square-Mesh Case. The cells are $E_{i,j} = [i, i+1] \times [j, j+1]$. Assume the solution is represented by degree k polynomial. The solution points on each square are formed by using tensor product; thus, there are $K = (k+1)^2$ of them. Denote by $\boldsymbol{u}_{i,j}$ the column vector of K solution values

$$\mathbf{u}_{i,j} = (u_{i,j,1,1},\ u_{i,j,1,2},\ ...,\ u_{i,j,1,k+1},\ \ u_{i,j,2,1},\ \ ...,\ u_{i,j,2,k+1},\ \ ...,\ u_{i,j,k+1,1},\ \ ...,\ u_{i,j,k+1,k+1})^T \tag{4.3}$$

where the superscript T denotes the transpose. The following property of the data plays a key role in the calculation of eigenvalues,

$$\mathbf{u}_{i-1,j} = e^{-I w_x} \mathbf{u}_{i,j} \quad \text{and} \quad \mathbf{u}_{i,j-1} = e^{-I w_y} \mathbf{u}_{i,j} \tag{4.4}$$

For the schemes of compact stencil discussed here, the solution can be expressed as

$$\frac{d\mathbf{u}_{i,j}}{dt} = \mathbf{C}_{0,0}\,\mathbf{u}_{i,j} + \mathbf{C}_{-1,0}\,\mathbf{u}_{i-1,j} + \mathbf{C}_{1,0}\,\mathbf{u}_{i+1,j} + \mathbf{C}_{0,-1}\,\mathbf{u}_{i,j-1} + \mathbf{C}_{0,1}\,\mathbf{u}_{i,j+1} \tag{4.5}$$

where all $\boldsymbol{C}_{\alpha,\beta}$ are $K \times K$ matrices. Using (4.4), we replace $\boldsymbol{u}_{i-1,j}$ by $e^{-I\,w_x}\mathbf{u}_{i,j}$, and similarly for all terms of the right hand side above. The result is

$$\frac{d\mathbf{u}_{i,j}}{dt} = \mathbf{S}\,\mathbf{u}_{i,j} \tag{4.6}$$

where

$$\mathbf{S} = \mathbf{C}_{0,0} + e^{-\mathbf{I}\,w_x}\,\mathbf{C}_{-1,0} + e^{\mathbf{I}\,w_x}\,\mathbf{C}_{1,0} + e^{-\mathbf{I}\,w_y}\,\mathbf{C}_{0,-1} + e^{\mathbf{I}\,w_y}\,\mathbf{C}_{0,1}. \tag{4.7}$$

Here, ***S***, which stands for ‘space’ or ‘semidiscrete’, is a $K \times K$ matrix with K eigenvalues. The one approximating $-(w_x{}^2 + w_y{}^2)$ is called the *principal eigenvalue* and is denoted by $S(w_x, w_y)$:

$$S(w_x, w_y) \approx -(w_x{}^2 + w_y{}^2). \tag{4.8}$$

All other eigenvalues are spurious. For stability, all eigenvalues must have nonpositive real parts.

It turns out that for all methods discussed, via Fourier analysis, the scheme on a square mesh and the corresponding 1D version have the same order of accuracy.

Concerning stability (time-stepping) limits, the method on a square mesh has a limit of 1/2 that of the 1D version. That is, the largest magnitude of all eigenvalues in the case of a square mesh is twice that of the corresponding 1D method. This reduction by a factor of 2 in the stability limit for the 2D diffusion equation is consistent with the reduction by a factor of $\sqrt{2}$ for advection equation in [13,14]: each derivative corresponds to a reduction by $\sqrt{2}$.

The minimum eigenvalues, orders of accuracy, and errors of the schemes on a square mesh are shown in Table 1. In the calculations of order of accuracy, the coarse mesh corresponds to $w_x = \pi/8$ and $w_y = \pi/10$, and the fine mesh, $w_x = \pi/16$ and $w_y = \pi/20$. Next, the principle eigenvalue $S\left(w_x, w_y\right)$ is compared to the exact eigenvalue $-\left(w_x^2 + w_y^2\right)$ to determine the error of the scheme. The errors *E*(*w*) and *E*(*w/2*) are then evaluated for the wave numbers of the coarse and fine meshes respectively. The order of accuracy is calculated by $\operatorname{Log}\left(\dfrac{E(w)}{E(w/2)}\right)\Big/\operatorname{Log}(2) - 2$ [14].

Note that all schemes are of order at least $2k$, i.e., they are super-convergent for $k \geq 2$. The advantage of the CDG scheme is that its accuracy order is $2(k+1)$; its disadvantage is that the minimum eigenvalues have rather large magnitudes (nearly three times) compared to those of the other schemes. Also note that for the square mesh case, the interior penalty (IP) scheme reduces to the BR2 scheme. As for the I-continuous scheme, the one employed here is the member whose triangular-mesh version is readily available, i.e., the member using the correction function g_{DG}, not the most accurate member, which employs the Legendre polynomial as correction function. Compared to BR2, the I-continuous scheme has a smaller error as well as a minimum eigenvalue with a smaller magnitude.

Triangular-Mesh Case. Each square $E_{i,j} = [i, i+1] \times [j, j+1]$ is cut into two triangles by the diagonal from the northwest to the southeast corners. To carry out the Fourier analysis for the triangular-mesh case, we must pair up these two triangles so that the solution vector on each square has a repeatable pattern explained below. Recall that if k is the degree of polynomial representing the solution, then the corresponding number of solution points on each triangle is $K=(k+2)(k+1)/2$. Combining the data on two triangles, denote the vector of $2K$ entries on the square $E_{i,j}$ by $\boldsymbol{u}_{i,j}$. The data $\boldsymbol{u}_{i,j}$ is repeatable in the sense that (4.4) holds. The semidiscrete equation is similar to (4.5) and, the corresponding $(2K)^2$ matrix ***S***, (4.7).

The minimum eigenvalues, orders of accuracy, and errors of the schemes on a triangular mesh are shown in Table 2. In the order of accuracy calculations, again, the coarse mesh corresponds to $w_x=\pi/8$ and $w_y=\pi/10$, and the fine mesh, $w_x=\pi/16$ and $w_y=\pi/20$.

Note that all schemes are of order $2k$. For a triangular mesh, the CDG scheme is no longer of order $2(k+1)$.It has the same order of accuracy as the other schemes, but its error is considerably smaller than those of the others. The disadvantage of the CDG scheme is that the minimum eigenvalues have rather large magnitudes (roughly three times) compared to those of the other schemes. Compared to BR2, the I-continuous scheme is slightly less accurate in this case; however, it has the advantage that its minimum eigenvalue has a slightly smaller magnitude.

Comparing Table 2 to 1, i.e., the schemes on a triangular mesh verses those on a square mesh, accuracy on a triangular mesh is better, but stability limit is worse. The reason is perhaps that corresponding to each k, there are more solution points on (the two triangles of) the square in the former case compared to the latter (i.e., $(k+2)(k+1)$ versus $(k+1)^2$).

As the last remark of this section, note that in practice, when the mesh is non-uniform and the equations are nonlinear, the super-accuracy (or super-convergence) property no longer holds. In fact, even for the case of a uniform mesh and linear equations, since the data is approximated by a polynomial of degree k in each cell, if we simply compare the numerical solution with the exact one, the degree of accuracy is only $k+1$; to observe super-accuracy, we may need to compare special quantities such as the average of the solutions on each square (even when the mesh is triangular). In short, super accuracy results do not hold in practice.

5. Numerical Results

5.1 Poisson's Equation

The following Poisson's equation

$$u_{xx} + u_{yy} = -\frac{\pi^2}{2}\sin\left(\frac{\pi x}{2}\right)\sin\left(\frac{\pi y}{2}\right), \tag{5.1}$$

with the exact solution $u = \sin\left(\frac{\pi x}{2}\right)\sin\left(\frac{\pi y}{2}\right)$ is solved on the domain [0, 1]X[0, 1] with both regular quadrilateral meshes and irregular triangular meshes (Figure 4). The solution variable u is fixed to the exact solution at the boundary faces. The CPR solver starts with the exact solution and marches forward in time with an LUSGS time integration [21] until the residual reaches machine zero.

The L_2 norm of the errors at the solution points is evaluated and the order of accuracy is calculated via an h-refinement by

$$p = \frac{\ln\left(\mathrm{Er}_1 / \mathrm{Er}_2\right)}{\ln\left(\sqrt{N_2 / N_1}\right)} \tag{5.2}$$

where Er_1 and Er_2 are the error norms for mesh 1 and mesh 2, with number of cells N_1 and N_2 respectively. (The error is always calculated in this manner in the rest of the paper).

Table 3 shows the accuracy results of the BR2, I-continuous and CDG approaches on quadrilateral meshes. All schemes produce comparable errors and orders of accuracy, with CDG bested the other two by a small margin. .

Table 4 shows accuracy results for irregular triangular meshes. Again, the errors by BR2, I-continuous and interior penalty schemes are comparable, and the CDG error is the best by a small margin

5.2 The Potential Flow over a Cylinder

The potential for an incompressible, irrotational and inviscid flow satisfies the Laplace equation (2.1). To test the CPR diffusion solver on curved boundaries, the potential flow over a cylinder with radius r=1 is computed. The exact solution for this case is $\varphi_{exact} = x\left(\frac{1}{x^2+y^2}+1\right)$.

The solution variable at the outer boundary faces is fixed to the exact solution. For the inner boundary, the Neumann condition $\vec{\nabla}\varphi\cdot\vec{n}=0$ is imposed. The curved boundary is always represented by polynomials of the same degree as that of the solution polynomials. The initial condition is set to the exact solution and an LUSGS solver is used with a convergence criteria of machine zero for all cases.

Figure 5a shows the 10x10 structured quadrilateral O-mesh used in this test case. Figure 5b shows the 10x10x2 triangular mesh, which is obtained by cut each quadrilateral cell into two triangles.

Figure 6 shows the solution φ by the BR2 method with p=2.

Tables 5 and 6 respectively show the numerical results for quadrilateral and triangular meshes. The L$_2$ errors produced by all the schemes seem comparable, and so is the order of accuracy.

Figure 7 shows the convergence history of the four schemes for p=3 on both quadrilateral and triangular meshes. The residuals are plotted against the iterations. Since the cost per iteration of the four approaches is about the same, these figures also represent the CPU time used. For both quadrilateral and triangular meshes, BR2 and interior penalty converge fastest and CDG, slowest.

5.3 The Couette Flow

Compressible Couette flow between two parallel walls is used to evaluate the accuracy of the method on irregular mixed grids. The exact solution for this case is

$$\begin{aligned} &u=\frac{U}{H}y,\ \ v=0 \\ &p=Const,\ \ \rho=\frac{p}{RT} \qquad . \\ &T=T_0+\frac{y}{H}(T_1-T_0)+\frac{\mu U^2}{2k}\frac{y}{H}\left(1-\frac{y}{H}\right) \end{aligned} \tag{5.3}$$

The following parameters are chosen: the speed of the upper wall $U=0.3$, the temperature of the lower wall $T_0=0.8$, the temperature of the upper wall $T_1=0.85$, viscosity $\mu=0.01$, domain height (in y direction) $H=2$, and the computational domain is a 4X2 rectangle.

The flow variables at boundary faces are fixed to the exact solution. All the tests cases presented below are obtained with an LUSGS implicit time integration approach [21], and all cases converge to machine zero. Density errors are used for accuracy evaluation.

Figure 8 shows the three meshes used in the mesh refinement study. The meshes are generated independently, rather than in an h-refinement manner. The schemes employed are 2nd-4th order accurate. The order of accuracy is also calculated by again Equation (5.2).

Table 7 shows accuracy results for the four schemes. Again, all the schemes are of comparable accuracy except for the case p=2 for CDG, where the order of accuracy is not quite as high as the others. The accuracy of CDG demonstrated in Fourier analysis and linear diffusion case is not carried over to this Navier-Stokes case. The BR2 and I-continuous approaches perform consistently well, whereas the interior penalty approach produces slightly larger errors for the case p=1. Note that although the orders of accuracy of all the schemes are slightly less than optimal, these results are obtained with highly irregular and mixed meshes of poor quality.

5.4 Laminar Flow around a NACA0012 Airfoil

Viscous laminar flow around a NACA 0012 airfoil is simulated with the CPR method, using the BR2, I-continuous, interior penalty and CDG schemes for the viscous flux. 2nd-5th order schemes are tested.

The flow conditions are Mach = 0.5 and Re = 5000, with an angle of attack of 1 degree. Under such conditions, steady laminar separations are expected for both upper and lower surfaces of the airfoil. Adiabatic no-slip wall condition is prescribed at the airfoil surface. Subsonic characteristic far field condition is used at the outer surface of the computational domain. The curved wall boundary is represented by the same degree polynomial as the solution. The computational domain extends 20 chord lengths away from the center of the airfoil.

Figure 9 shows the mesh of 2692 cells, which is composed of regular quadrilateral elements near the airfoil and irregular mixed elements elsewhere, with some refinement at the trailing edge. A LUSGS solver is used for time integration and all cases converge to machine zero.

Figure 10 (a-c) shows the computed Mach number contours of 2nd – 4th order schemes. Only the BR2 results are shown, since the results of other schemes are very similar. Due to the coarse mesh, the 2nd order results are not smooth, especially at the wake. Note that for the 3rd and 4th order cases, the contour lines are smooth across the interfaces between regular cells and irregular ones and also between triangular cells and quadrilateral ones. Visually, 3rd and 4th order results are nearly identical.

The skin friction coefficient C_f is defined as

$$C_f = \frac{\tau_w}{\frac{1}{2}\rho U_\infty^2} \tag{5.4}$$

where the shear stress at wall is computed by $|\tau_w| = \mu \frac{\partial V}{\partial n}$, with its sign chosen to be the same as $\frac{\partial u}{\partial n}$.

Figure 11 shows the C_f distribution near the separation point. For the four diffusion schemes, the 3rd and 4th order results are very close; this fact indicates convergence with *p*-refinement. Except for the 2nd order case, the four schemes produce almost identical distributions of C_f.

Tables 8-10 show the computed pressure drag coefficients $C_{D,p}$, friction drag coefficients $C_{D,f}$, and the separation points on the upper wall. The data converge as the polynomial degree increases and, again, there is no significant difference among the four schemes considered.

Figures 12(a, b) show the convergence history of the 4th order case, which provides an estimate of the efficiency of the four diffusion schemes. The same time step, which is also the largest allowable by stability, is used for all schemes. In terms of both iteration and CPU time, CDG is least efficient, while the other three approaches show no significant difference.

5.5 Unsteady Flow around a Circular Cylinder

The unsteady laminar flow around a cylinder is simulated with the 2nd – 4th order CPR methods using the BR2 diffusion scheme. The Reynolds number *Re = 75*, the free stream Mach number *M = 0.2*. A vortex street is expected to form in the wake of the cylinder. The frequency of the vortex shedding is often called the Strouhal number defined by

$$St = \frac{f_s L_c}{U_\infty}. \tag{5.5}$$

The length scale L_c here is the diameter of the cylinder. In an experimental study by Williamson [30], *St* was found to be 0.148.

Subsonic far field boundary condition is used at the outer boundary of the domain. Adiabatic wall condition is used for the cylinder surface. A 2nd order unsteady LUSGS solver is used for time integration.

Figure 13 shows the mesh, which contains 2,028 cells, with regular quadrilateral cells near the cylinder and in the wake region, and irregular mixed cells elsewhere. The effects of domain size, time step and convergence criteria of inner iterations are carefully studied to make sure the correct numerical solution is obtained.

Figure 14 (a-c) shows the instantaneous Mach number contours. Note that the 2nd order contours in (a) are not smooth. The 3rd and 4th order results are smooth and similar, with 4th order contours smoother at 20 diameters downstream.

Table 11 shows the average drag coefficient $\overline{C_D}$ and computed *St* for *p*=1-3. Both converge as *p* increases, and the computed *St* is comparable to the experimental data.

6. Conclusions

The CPR formulation, which has the advantage of simplicity and economy, is extended to the diffusion and Navier-stokes equations on hybrid meshes of triangles and quadrilaterals. The four schemes of compact stencil employed to discretize the diffusion terms are BR2, interior penalty, compact DG (CDG), and I-continuous. The first three of these schemes, originally derived using the integral formulation, were recast here in the CPR framework, whereas the I-continuous scheme, originally derived for a quadrilateral mesh, was extended to a triangular mesh. Fourier stability and accuracy analyses on quadrilateral and triangular meshes were carried out. Numerical tests varying from linear diffusion to steady and unsteady laminar flows were conducted.

The results show that the CPR approach produces highly accurate schemes of expected order of accuracy, i.e., schemes using polynomial of degree k is accurate to order k+1. Concerning methods for diffusion, the four compact schemes are comparable in terms of accuracy. In terms of convergence rate, however, the CDG scheme is slowest: this finding by numerical experiments is consistent with the finding via Fourier analysis. Among the four schemes, due to its simplicity, BR2 is an optimal choice; while I-continuous approach is promising, a more accurate triangular-mesh version must be found for the scheme to be competitive; the interior penalty approach is the simplest, but is also slightly less accurate. Finally, the CPR formulation presented here can be extended to 3D and, due to its hybrid-mesh capability, the resulting methods can be applied to practical problems involving complex geometries.

Acknowledgments

The first two authors were funded by AFOSR grant FA9550-06-1-0146. The third author is supported by NASA’s Fundamental Aeronautics Program. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of AFOSR, NASA or the U.S. Government.

References

[1] D.N. Arnold, F. Brezzi, B. Cockburn and L.D. Marini, Unified analysis of discontinuous Galerkin methods for elliptic problems, SIAM J. Numer. Anal., 19(4): 742-760, 2002.

[2] F. Bassi and S. Rebay, A high-order accurate discontinuous finite element method for the numerical solution of the compressible Navier–Stokes equations. J Comp Phys 1997;131(1):267–79.

[3] F. Bassi and S. Rebay, High-order accurate discontinuous finite element solution of the 2D Euler equations, J. Comput. Phys. 138, 251-285 (1997).

[4] F.Bassi and S. Rebay, GMRES Discontinuous Galerkin Solution of the Compressible Navier-Stokes Equations. In Karniadakis Cockburn and Shu, editors, 208. Springer, Berlin, 2000.

[5] B. Cockburn and C.-W. Shu, TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws II: general framework, Mathematics of Computation 52, 411-435 (1989).

[6] B. Cockburn, S.-Y. Lin and C.-W. Shu, TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws III: one-dimensional systems, J. Comput. Phys. 84, 90-113 (1989).

[7] B. Cockburn and C.-W. Shu, The Runge-Kutta discontinuous Garlerkin method for conservation laws V: multidimensional systems, J. Comput. Phys., 141, 199 - 224, (1998).

[8] B. Cockburn and C.-W. Shu, The local discontinuous Galerkin methods for time-dependent convection diffusion systems, SIAM J. Numer. Anal., 35(1998), pp. 2440-2463.

[9] V. Dolejší, On the discontinuous Galerkin method for numerical solution of the Navier-Stokes equations, Int. J. Numer. Methods Fluids, 45:1083-1106, 2004.

[10] H. Gao and Z.J. Wang , A High-Order Lifting Collocation Penalty Formulation for the Navier-Stokes Equations on 2-D Mixed Grids, AIAA-2009-3784.

[11] T. Haga, H. Gao and Z.J. Wang, A High-Order Unifying Discontinuous Formulation for 3-D Mixed Grids, AIAA-2010-540.

[12] R. Hartmann and P. Houston, Symmetric interior penalty DG methods for the incompressible Navier-Stokes Equations I: Method formulation. Int. J. Numer. Anal. Model., 1:1-20.

[13] H.T. Huynh, A flux reconstruction approach to high-order schemes including discontinuous Galerkin methods, AIAA Paper 2007-4079.

[14] H.T. Huynh, A Reconstruction Approach to High-Order Schemes Including Discontinuous Galerkin for Diffusion, AIAA Paper 2009-403.

[15] D.A. Kopriva and J.H. Kolias, A conservative staggered-grid Chebyshev multidomain method for compressible flows, J. Comp. Physics, 125(1996), pp. 244-261.

[16] C. Liang, A. Jameson and Z. J. Wang, Spectral Difference method for two-dimensional compressible flow on unstructured grids with mixed elements, Journal of Computational Physics, vol 228, pp 2847-2858, 2009.

[17] Y. Liu, Vinokur M, and Wang ZJ. "Discontinuous Spectral Difference Method for Conservation Laws on Unstructured Grids," in Proceedings of the 3rd International Conference on Computational Fluid Dynamics, Toronto, Canada, July 12-16 2004.

[18] Y. Liu, Vinokur M, and Wang ZJ. "Discontinuous Spectral Difference Method for Conservation Laws on Unstructured Grids," Journal of Computational Physics Vol. 216, pp. 780-801 (2006).

[19] G. May and A. Jameson, “A spectral difference method for the Euler and Navier-Stokes equations”, AIAA paper No. 2006-304, 2006.

[20] J. Peraire and P.-O. Persson, The compact discontinuous Galerkin (CDG) method for elliptic problems, SIAM J. Sci. Comput., Vol. 30, No. 4 (2008) pp. 1806-1824.

[21] Y. Sun, Z.J. Wang, and Y. Liu, Efficient implicit non-linear LU-SGS approach for compressible flow computation using high-order spectral difference method, Commun. Comput. Phys. Vol. 5, No. 2-4, pp. 760-778.

[22] K. Van den Abeele, C. Lacor, and Z. J. Wang. On the stability and accuracy of the spectral difference method. J. Sci. Comput., 37(2):162-188, 2008.

[23] B. van Leer and S. Nomura, Discontinuous Galerkin for Diffusion, AIAA Paper 2005-5108.

[24] Z.J. Wang, Spectral (Finite) volume method for conservation laws on unstructured grids: basic formulation, J. Comput. Phys. Vol. 178, pp. 210-251, 2002.

[25] Z.J. Wang, High-order methods for the Euler and Navier-Stokes equations on unstructured grids, Journal of Progress in Aerospace Sciences, Vol. 43, pp. 1-47, 2007.

[26] Z.J. Wang and Y. Liu, Spectral (finite) volume method for conservation laws on unstructured grids iii: one-dimensional systems and partition optimization, Journal of Scientific Computing, Vol. 20, No. 1, pp. 137-157 (2004).

[27] Z.J. Wang and Y. Liu, Spectral (finite) volume method for conservation laws on unstructured grids iii: one-dimensional systems and partition optimization, Journal of Scientific Computing, Vol. 20, No. 1, pp. 137-157 (2004).

[28] Z.J. Wang, L. Zhang and Y. Liu, Spectral (Finite) Volume Method for Conservation Laws on Unstructured Grids IV: Extension to Two-Dimensional Euler Equations, Journal of Computational Physics Vol. 194, No. 2, pp. 716-741 (2004).

[29] Z.J. Wang and H. Gao, A unifying lifting collocation penalty formulation including the discontinuous Galerkin, spectral volume/difference methods for conservation laws on mixed grids, Journal of Computational Physics 228 (2009) 8161–8186.

[30] C. H. K. Williamson. Oblique and parallel modes of vortex shedding in the wake of a cylinder at low Reynolds number, J. Fluid Mech., 206:579–627, 1989.

Table 1. Minimum eigenvalues, orders of accuracy, and errors for schemes on a square mesh.

Square-mesh case $K = 2$	Min. Eigval.	Ord. Acc.	Coarse mesh error, $w_x = \pi/8$, $w_y = \pi/10$	Fine mesh error, $w_x = \pi/16$, $w_y = \pi/20$
1. BR2	− 26.7	1.96	-2.68e-03	1.73e-04
2. I-Continuous	− 24.0	2.00	1.39e-03	8.72e-05
3. IP = BR2				
4. CDG	− 72.	4.01	8.64e-06	1.34e-07

$K = 3$				
1. BR2	− 120.	3.98	2.72e-06	4.29e-08
2. I-Continuous	− 84.	4.00	2.76e-06	4.31e-08
3. IP = BR2				
4. CDG	− 297.	6.00	5.25e-09	2.06e-11

$K = 4$				
1. BR2	− 340.	5.89	-5.94e-09	-2.50e-11
2. I-Continuous	− 244.	6.00	2.62e-09	1.02e-11
3. IP = BR2				
4. CDG	− 878.	8.00	2.17e-12	2.12e-15

Table 2. Minimum eigenvalues, orders of accuracy, and errors for schemes on a triangular mesh.

Triangular-mesh case $K = 2$	Min. Eigval.	Ord. Acc.	Coarse mesh error, $w_x = \pi/8$, $w_y = \pi/10$	Fine mesh error, $w_x = \pi/16$, $w_y = \pi/20$
1. BR2	− 66.7	1.97	-5.83e-04	-3.72e-05
2. I-Continuous	− 56.8	2.00	8.90e-04	5.55e-05
3. IP	− 60.	2.00	8.45e-04	5.27e-05
4. CDG	− 160.	1.84	-1.14e-04	-1.63e-05

$K = 3$				
1. BR2	− 204.	3.99	1.00e-06	1.57e-08
2. I-Continuous	− 203.	4.00	1.64e-06	2.57e-08
3. IP	− 207.	4.01	3.42e-06	5.30e-08
4. CDG	− 565.	3.96	-4.19e-08	-6.74e-10

$K = 4$				
1. BR2	− 522.	5.95	-1.32e-09	-5.33e-12
2. I-Continuous	− 508.	6.00	1.42e-09	5.54e-12
3. IP	− 516.	5.82	-1.47e-09	-6.51e-12
4. CDG	− 1458.	5.98	-4.10e-11	-1.62e-13

Table 3. Accuracy results of Poisson's equation with regular quadrilateral mesh (Interior Penalty is identical to BR2).

Polynomial Degree	Mesh	BR2		I-Continuous		CDG	
		L2 error	L2 order	L2 error	L2 Order	L2 error	L2 Order
P=2	5x5	1.949e-04	-	1.915e-04	-	1.329e-04	-
	10x10	2.360e-05	3.05	2.337e-05	3.03	1.718e-05	2.95
	20x20	2.891e-06	3.03	2.876e-06	3.02	2.176e-06	2.98
	40x40	3.573e-07	3.02	3.564e-07	3.01	2.735e-07	2.99
P=3	5x5	4.396e-06	-	5.179e-06	-	2.738e-06	-
	10x10	2.438e-07	4.17	3.298e-07	3.97	1.777e-07	3.95
	20x20	1.409e-08	4.11	2.080e-08	3.99	1.132e-08	3.97
	40x40	8.408e-10	4.07	1.306e-09	3.99	7.140e-10	3.99
P=4	5x5	6.501e-08	-	6.385e-08	-	4.398e-08	-
	10x10	1.981e-09	5.04	1.961e-09	5.03	1.398e-09	4.98
	20x20	6.108e-11	5.02	6.076e-11	5.01	4.398e-11	4.99

Table 4. Accuracy results of Poisson's equation with irregular triangular mesh.

Polynomial Degree	# of cells	BR-2		I-continuous	
		L2 error	L2 order	L2 error	L2 order
P=2	58	2.993e-04	-	2.995e-04	-
	230	4.188e-05	2.85	4.130e-05	2.88
	936	4.590e-06	3.15	4.624e-06	3.12
P=3	58	1.196e-05	-	1.238e-05	-
	230	5.887e-07	4.37	6.476e-07	4.28
	936	3.382e-08	4.07	3.722e-08	4.07
Polynomial Degree	# of cells	Interior Penalty		CDG	
		L2 error	L2 order	L2 error	L2 order
P=2	58	4.026e-04	-	2.308e-04	-
	230	5.590e-05	2.87	3.343e-05	2.80
	936	6.226e-06	3.13	3.832e-06	3.09
P=3	58	1.509e-05	-	9.185e-06	-
	230	7.428e-07	4.37	4.578e-07	4.35
	936	4.225e-08	4.09	2.812e-08	3.98

Table 5. Accuracy results for a potential flow over a cylinder with quadrilateral meshes (Interior Penalty is identical to BR2).

Polynomial Degree	Mesh	BR2		I-continuous		CDG	
		L_2 error	L_2 order	L_2 error	L_2 Order	L_2 error	L_2 Order
P=2	5x5	3.610e-03	-	3.759e-03	-	3.103e-03	-
	10x10	4.653e-04	2.96	4.721e-04	2.99	4.125e-04	2.91
	20x20	6.061e-05	2.94	6.013e-05	2.97	5.331e-05	2.95
P=3	5x5	1.166e-04	-	6.231e-04	-	4.459e-04	-
	10x10	4.131e-06	4.82	3.750e-05	4.05	3.148e-05	3.82
	20x20	4.246e-07	3.28	2.394e-06	3.97	2.114e-06	3.90
P=4	5x5	8.845e-05	-	9.185e-05	-	7.535e-05	-
	10x10	2.847e-06	4.96	2.873e-06	5.00	2.605e-06	4.85
	20x20	9.264e-08	4.94	9.264e-08	4.95	8.627e-08	4.92

Table 6. Accuracy results for a potential flow over a cylinder with triangular meshes.

Polynomial Degree	Mesh	BR-2		I-continuous	
		L_2 error	L_2 order	L_2 error	L_2 order
P=2	10x10x2	9.485e-04	-	7.935e-04	-
	20x20x2	9.779e-05	3.28	7.776e-05	3.35
	40x40x2	1.179e-05	3.05	9.404e-06	3.05
P=3	10x10x2	6.881e-05	-	7.764e-05	-
	20x20x2	3.944e-06	4.12	3.651e-06	4.41
	40x40x2	2.440e-07	4.01	2.219e-07	4.04
Polynomial Degree	Mesh	Interior Penalty		CDG	
		L2 error	L2 order	L2 error	L2 order
P=2	10x10x2	1.976e-03	-	7.564e-04	-
	20x20x2	1.301e-04	3.92	7.863e-05	3.27
	40x40x2	1.477e-05	3.14	9.229e-06	3.09
P=3	10x10x2	9.140e-05	-	5.199e-05	-
	20x20x2	4.746e-06	4.27	3.022e-06	4.10
	40x40x2	2.956e-07	4.00	1.918e-07	3.98

Table 7. Accuracy results of Couette flow with hybrid mesh.

Polynomial Degree & Mesh (No. of cells)		BR2		I-continuous	
		L_2 Error	L_2 Order	L_2 Error	L_2 Order
p=1	61	7.261e-05	-	7.112e-05	-
	250	2.312e-05	1.62	2.306e-05	1.60
	919	7.151e-06	1.80	7.162e-06	1.80
p=2	61	8.914e-07	-	8.795e-07	-
	250	1.309e-07	2.72	1.400e-07	2.61
	919	2.091e-08	2.82	2.456e-08	2.67
p=3	61	8.525e-09	-	8.190e-09	-
	250	6.525e-10	3.64	6.157e-10	3.67
	919	5.785e-11	3.72	5.186e-11	3.80
Polynomial Degree & Mesh (No. of cells)		Interior Penalty		CDG	
		L_2 Error	L_2 Order	L_2 Error	L_2 Order
p=1	61	1.007e-04	-	1.019e-04	-
	250	3.421e-05	1.53	3.363e-05	1.57
	919	1.013e-05	1.87	9.714e-06	1.91
p=2	61	8.499e-07	-	8.766e-07	-
	250	1.278e-07	2.69	1.696e-07	2.33
	919	2.045e-08	2.82	3.641e-08	2.36
p=3	61	9.444e-09	-	8.834e-09	-
	250	7.044e-10	3.68	7.555e-10	3.49
	919	6.158e-11	3.74	7.228e-11	3.61

Table 8. Flow over a NACA0012 airfoil—pressure drag coefficients CD,p

	BR2	I-Continuous	Interior Penalty	CDG
p=1	0.02252	0.02233	0.02252	0.02173
p=2	0.02275	0.02272	0.02276	0.02259
p=3	0.02295	0.02294	0.02295	0.02294
p=4	0.02296	0.02295	0.02296	0.02295

Table 9. Flow over a NACA0012 airfoil—friction drag coefficients CD,f

	BR2	I-Continuous	Interior Penalty	CDG
p=1	0.03248	0.03289	0.03248	0.03287
p=2	0.03271	0.03272	0.03271	0.03277
p=3	0.03250	0.03250	0.03251	0.03250
p=4	0.03248	0.03250	0.03249	0.03248

Table 10. Flow over a NACA0012 airfoil—separation point on the upper surface.

	BR2	I-Continuous	Interior Penalty	CDG
p=1	79.84%	78.30%	79.84%	80.39%
p=2	68.67%	68.65%	68.67%	68.67%
p=3	67.84%	67.81%	67.84%	67.82%
p=4	67.91%	67.90%	67.91%	67.91%

Table 11. P-refinement study for unsteady flow around a cylinder.

Viscous Method -- Polynomial Degree	$\overline{C_D}$	St
BR2--1	1.4299	0.153
BR2--2	1.4365	0.153
BR2--3	1.4364	0.153

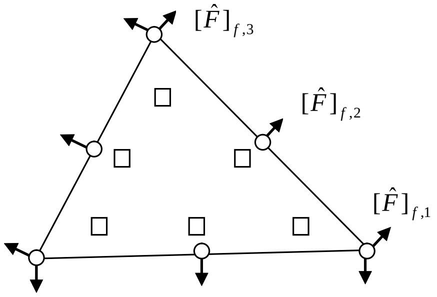


Figure 1. Solution points (squares) and flux points (circles) for k = 2.

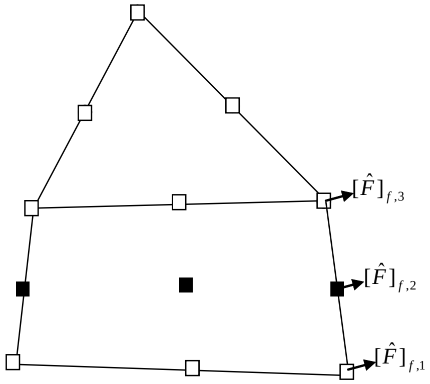


Figure 2. Solution points for the 3rd order CPR scheme on hybrid meshes.

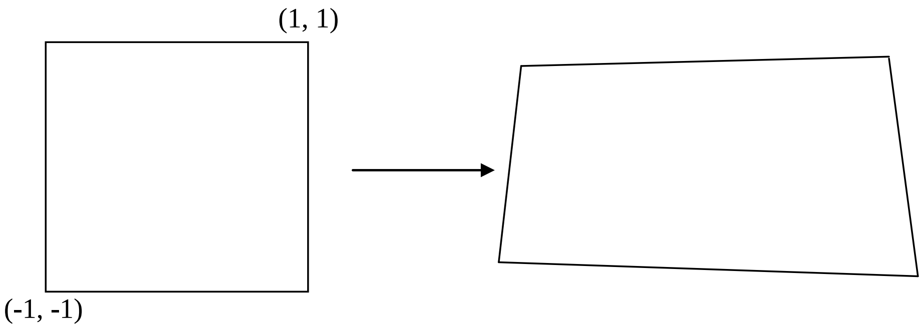


Figure 3. Transformation of a quadrilateral element to a standard element.

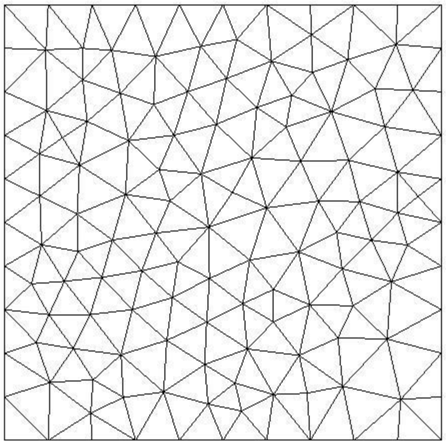
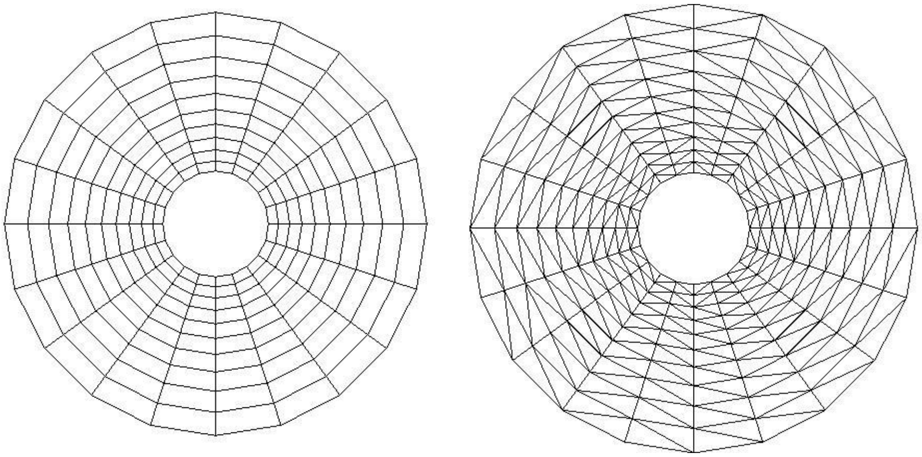


Figure 4. Irregular triangular mesh for Poisson's equation on [0,1] X[0,1].



(a) Quadrilateral mesh **(b) Triangular mesh**

Figure 5. Meshes for potential flow over a cylinder.

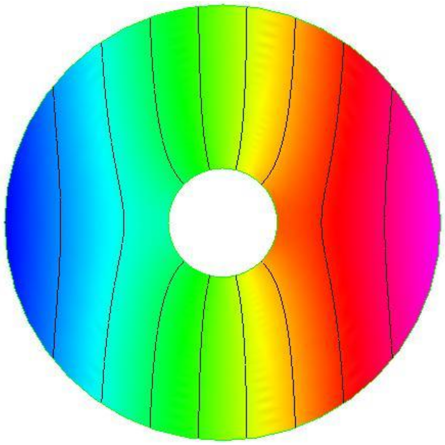
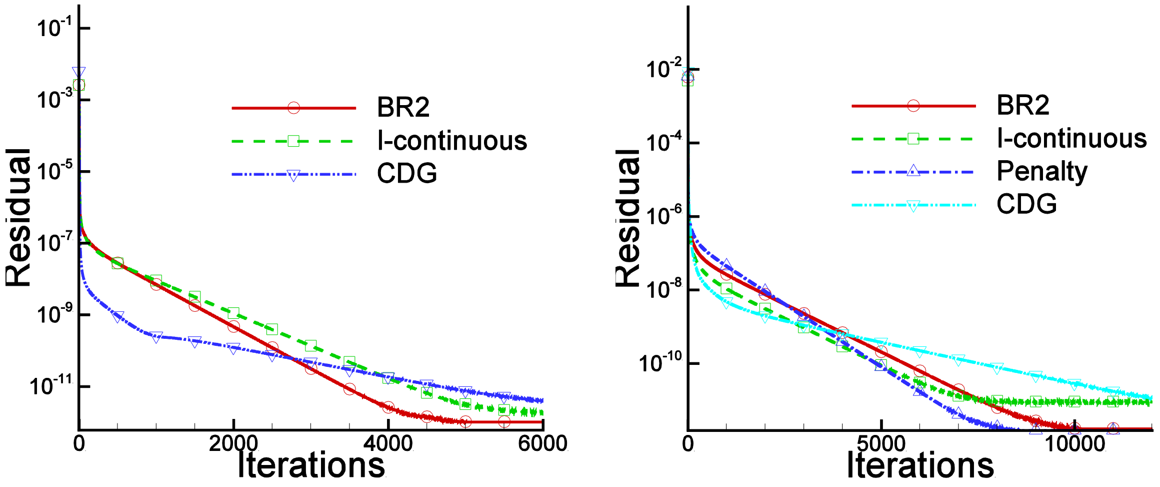
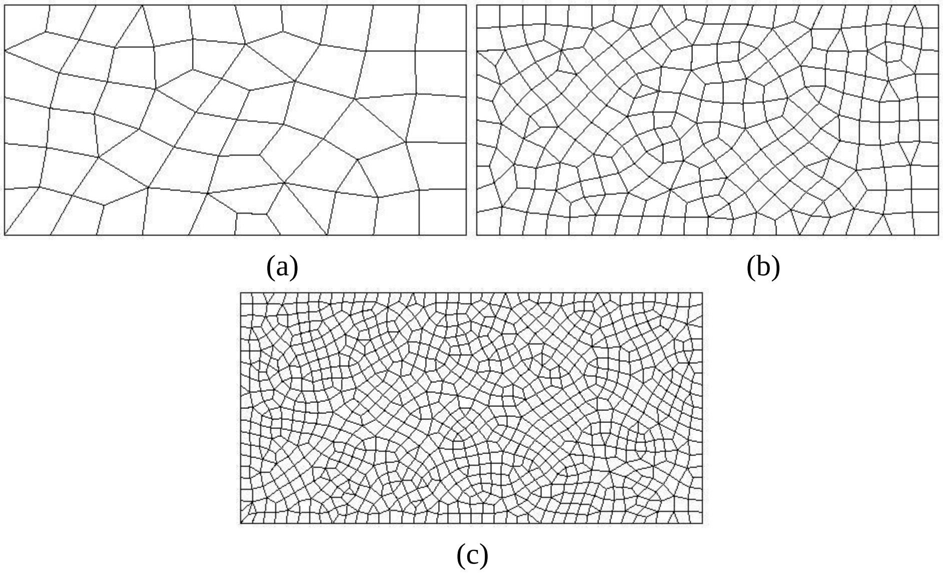


Figure 6. Solution contours for potential flow over a cylinder (BR2, 3rd order, 20X20 mesh).



(a) **(b)**

Figure 7. Convergence history of potential flow over a cylinder (4th order, 40x40 meshes) (a) quadrilateral mesh (b) triangular mesh.



(a) (b)

(c)

Figure 8. Mixed mesh used for Couette flow (a) 10X5 with 61 cells (b) 20X10 with 250 cells (c) 40X20 with 919 cells.

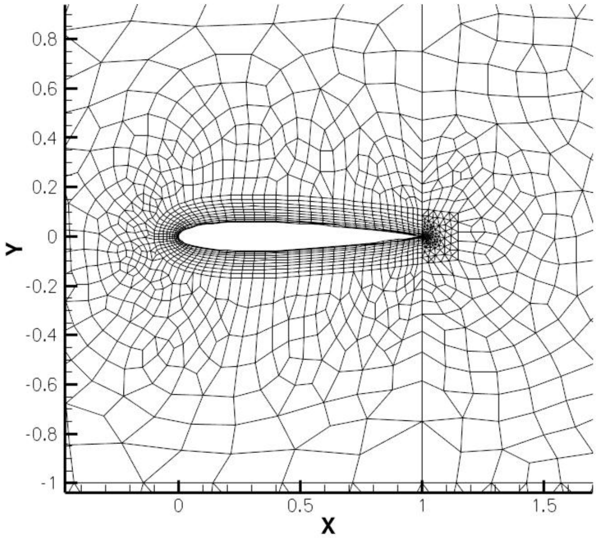
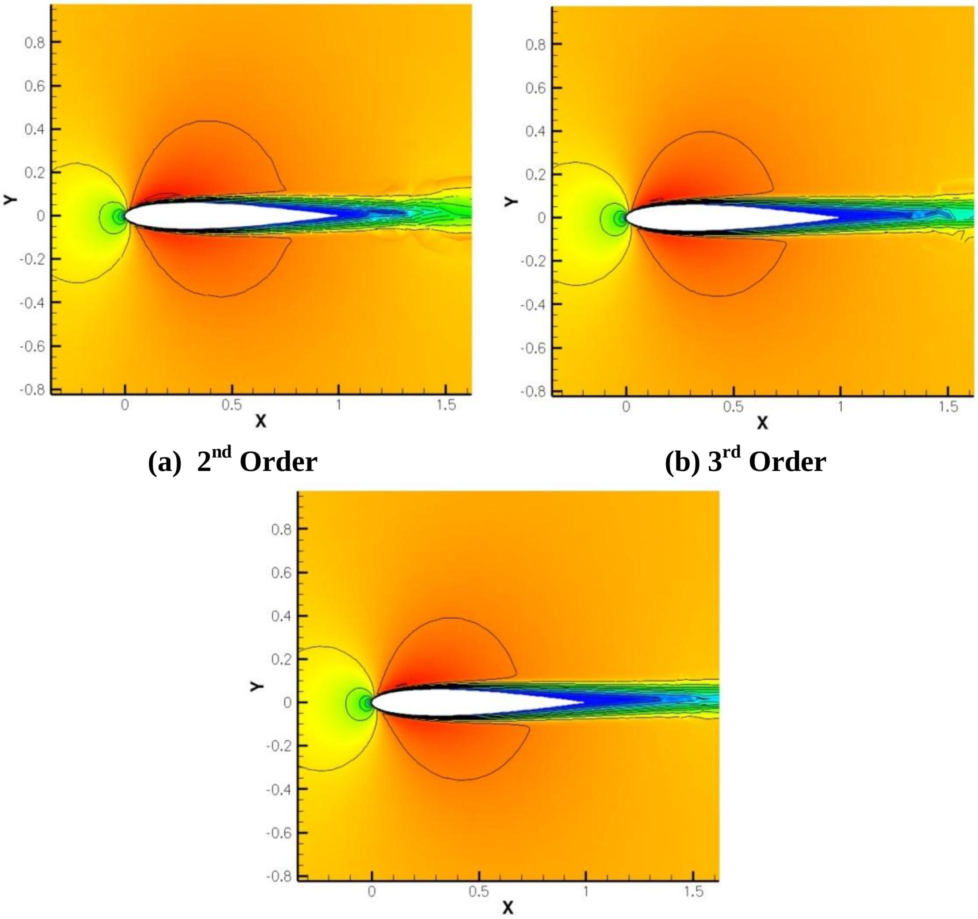


Figure 9. Mixed mesh around an NACA0012 airfoil.



(a) 2nd Order **(b) 3rd Order**

(c) 4th Order

Figure 10. Mach number contours of flow around an NACA 0012 airfoil.

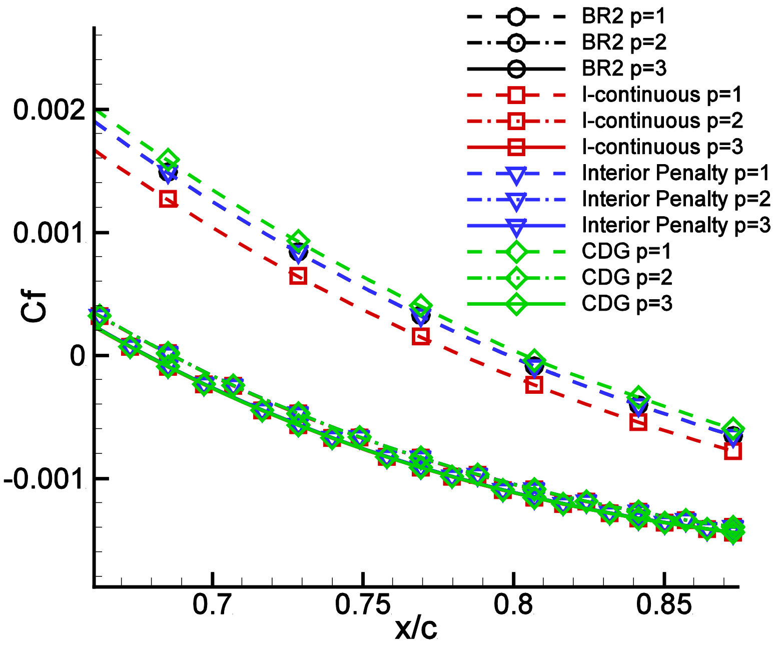
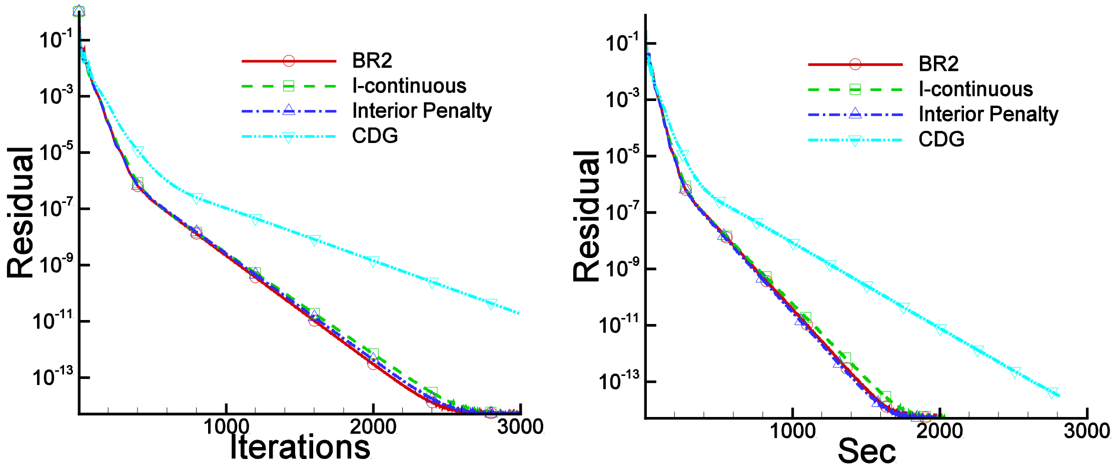
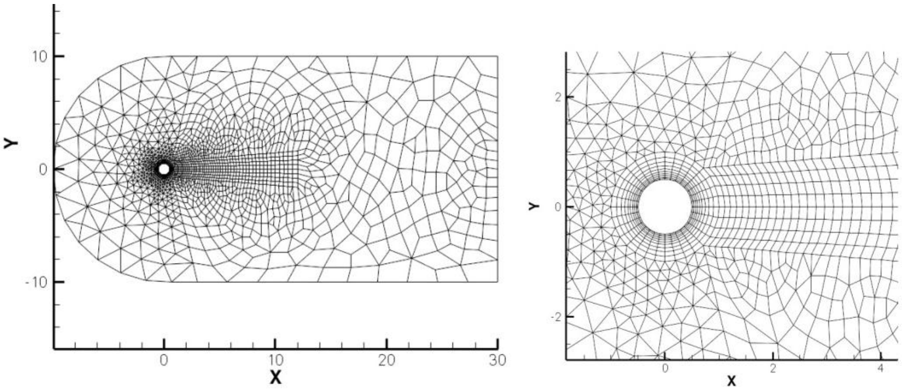


Figure 11. C_f on the upper surface of NACA0012 airfoil for BR2, I-continuous, interior penalty and CDG.



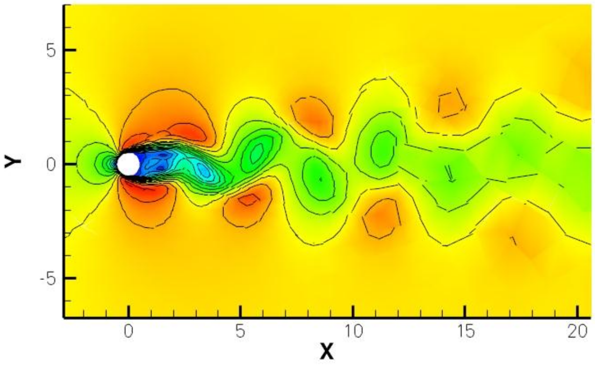
(a) **(b)**

Figure 12. 4th order convergence history of laminar flow over NACA0012 airfoil in terms of (a) iterations (b) CPU time.

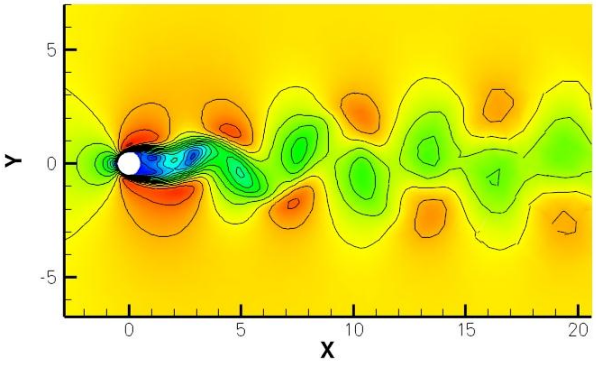


(a) The whole domain **(b) near the cylinder**

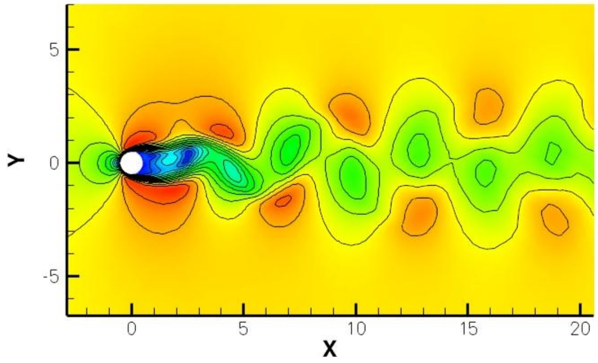
Figure 13. Mixed mesh for unsteady flow around a cylinder.



(a) 2nd order



(b) 3rd order



(c) 4th order

Figure 14. Mach number contours for unsteady flow around a cylinder.

CHAPTER 4. A Study of Curved Boundary Representations for 2D High-order Euler Solvers

A paper published in the Journal of Scientific Computing

Haiyang Gao, Z.J. Wang and Yen Liu

Abstract

The present study discusses several approaches of representing curve boundaries with high-order Euler solvers. The aim is to minimize the solution entropy error. The traditional boundary representation based on interpolation polynomials is presented first. A new boundary representation based on the Bezier curve is then developed and analyzed. Several numerical tests are conducted to compare both representations. The new approach is shown to have several advantages for complicated geometries.

1. Introduction

The last two decades has witnessed the emergence of several high-order methods, such as the discontinuous Galerkin (DG) [1-4], spectral volume (SV) [10-12], spectral difference (SD) [11,15], and the recently developed flux reconstruction (FR) [8] and lifting collocation penalty (LCP) methods [16]. All the above methods use piecewise polynomials to approximate the solution within an element, and thus result in compact spatial discretizations.

As high-order methods become increasingly mature for real world applications, the high-order boundary treatment becomes a critical issue. In mainstream finite volume solvers, the wall boundary of the geometry is usually represented by piecewise straight segments or planar facets. However, this linear boundary representation is far from sufficient for high-order methods. When the error from the geometry dominates the discretization error, the benefit of high-order methods is largely lost. Thus, the curvilinear element was introduced by Gordon and Hall [6, 7] to solve the problem, which was also used by Kopriva [9] in the staggered-grid method. Bassi and Rebay [1, 2] also identified the importance of geometry representation while solving the Euler equations with the DG method, and they demonstrated that high order boundary was mandatory if any meaningful solution was to be obtained. Besides, the order of boundary representation should be as high as the order of the solution reconstruction to achieve an optimal rate of convergence [1].

When developing the SV method for the Euler equations, Wang and Liu [14] also addressed the importance of high order boundary representation. In their numerical tests, it was found that higher than quadratic boundary representation is necessary to obtain any physically acceptable solutions.

In the above implementations of high order boundary, a polynomial geometric mapping is used to map a curve boundary element in the physical domain to a standard element in the computational domain.

Another approach was proposed by Krivodonova and Berger [10]. Instead of resorting to a full-fledged non-linear transformation, the ghost state is computed with interpolated normal directions at the quadrature points. After that, a Riemann problem is solved along the normal directions of straight boundary edges to get the flux at the solid wall boundary. The approach was very simple yet produced smooth and physically correct solutions.

The present study focuses on the geometric mapping technique with a reflection solid wall boundary condition. In addition to the usual approach of interpolation mapping, a new mapping approach based on the Bezier curve is developed and examined with numerical tests. We are especially interested in the performance of these curve boundary methods for meshes where the boundary is only represented by straight segments. Because in most cases with complicated geometry, these are what most of today's mesh generators can generate.

The numerical tests in this study are conducted with the newly developed LCP solver [16], which is capable of solving the 2D Euler equations on a mixed unstructured mesh. In addition, the LCP method is a differential formulation without any integral quadratures, which makes it easy for any mapping to be implemented. However, the non-linear mapping used here is not limited to the LCP method and can be easily applied to other high order methods.

2. Interpolation Polynomial Mapping (IP mapping)

To map a curved boundary element to a standard element, either a triangle or quadrilateral, we consider a one to one transformation from the physical domain (x, y) to the computational domain (ξ,η),

$$x = x(\xi,\eta) \tag{2.1}$$

$$y = y(\xi,\eta) \tag{2.2}$$

Following the usual practice in the finite element method [15], isoparametric elements can be used for this purpose, in the form of

$$\vec{r} = \sum_{j=1}^{m} M_j(\xi,\eta)\vec{r}_j \tag{2.3}$$

where $\vec{r} = (x, y)$, $M_j(\xi,\eta)$ are the shape functions and $\vec{r}_j$ is a set of interpolation points on the curved wall boundary.

A quadratic triangular isoparametric element requires the coordinates of 6 nodes and a cubic element requires 10 nodes. But in most 2D curve wall boundary cells, only one side is curved. This allows for a simplified curved wall element as shown in Figure 1, using only 4 nodes for a quadratic and 5 nodes for a cubic triangular element [14].

In the simplified curve boundary elements, the curved side is represented by an interpolation polynomial of degree *n* (called *P-n* boundary from here on), while the other sides are left straight. For simplicity, all the test cases in the present study only consider these simplified curved boundary cells. More general high-order parametric elements can be implemented similarly.

The IP mapping provides an accurate representation of the curve boundary, on the condition that the interpolating nodes are accurate. However, for complicated real world geometry, the nodes are normally obtained by interpolation. This may cause additional errors, since the IP mapping is found to be rather sensitive to the geometric accuracy of the interpolating nodes in numerical tests presented later.

Another drawback of the IP mapping is that the curved wall is only C^0 continuous between different elements, i.e., the tangent of the curve is only piecewise continuous. This causes sharp corners to form in regions with large curvature.

3. Bezier Curve Based Geometry Mapping (Bezier Mapping)

Instead of approximating the curve boundary by interpolating a set of interior nodes, which are located on the geometry, another option is to find a polynomial that can ensure

C^1 continuity between segments. The Bezier curve is a common way to achieve this goal [6].

The Bezier curve is a polynomial that is defined by two end points and several interior control points depending on the order of the curve. It does not usually go through the interior control points, but the tangent directions at both ends of the curve can be specified by those interior points, as shown in Figure 2 for Bezier curved of various degrees.

If the exact geometry is known, the exact normal can be specified for each node. Otherwise, an average normal of the two adjacent linear segments can be used to approximate the true normal.

3.1 Geometric Mapping Based on a Quadratic Bezier Curve

A quadratic Bezier curve is the path traced by function $\vec{B}(t)$ representing the position vector, given control points $\vec{P}_0$, $\vec{P}_1$, and $\vec{P}_2$,

$$\vec{B}(t)=(1-t)^2\vec{P}_0+2(1-t)t\vec{P}_1+t^2\vec{P}_2,\quad t\in[0,1], \tag{3.1}$$

It has the property that the tangent direction is $\overrightarrow{P_0P_1}$ at point $\vec{P}_0$, and $\overrightarrow{P_1P_2}$ at point $\vec{P}_2$. We can extend both tangent directions at the two end points and find the intersection point $\vec{P}_1$ (assuming there is an intersection for now), which can then be used as the control point to build a Bezier curve as shown in Figure 3. Let the unit tangent vectors be $\vec{t}_1$ and $\vec{t}_2$. The intersection can be found to be

$$\vec{P}_1=\vec{P}_0+\frac{\left|(\vec{P}_2-\vec{P}_0)\times\vec{t}_2\right|}{\left|\vec{t}_1\times\vec{t}_2\right|}\vec{t}_1. \tag{3.2}$$

Obviously, a necessary and sufficient condition for the existence of an intersection is that $\vec{t}_1$ and $\vec{t}_2$ not be parallel to each other. Given the two end points and two tangent directions, the Bezier curve is then completely defined.

Here only the mapping for a quadrilateral cell shown in Figure 4 is demonstrated, and the mapping for a triangular cell can be obtained in a similar manner. The mapping is in the form of Eqn. (2.1-2).

Since $0\le t\le 1$, first, the Bezier curve must be expressed with $\xi\in[-1,1]$, using $t=\dfrac{\xi+1}{2}$

$$\vec{B}(\xi)=\vec{C}_{b,0}+\vec{C}_{b,1}\xi+\vec{C}_{b,2}\xi^2 \tag{3.3}$$

where $\vec{C}_{b,0}=\dfrac{\vec{P}_0+2\vec{P}_1+\vec{P}_2}{4},\vec{C}_{b,1}=\dfrac{-\vec{P}_0+\vec{P}_2}{2},\vec{C}_{b,2}=\dfrac{\vec{P}_0-2\vec{P}_1+\vec{P}_2}{4}$.

Here, *Edge 02* in the physical domain is the Bezier curve just obtained. To obtain this transformation, the following basis functions are selected

$$L=\begin{bmatrix}1 & \xi & \eta & \xi\eta & \xi^2(1-\eta)\end{bmatrix}. \tag{3.4}$$

The transformation can be expressed in this form,

$$\vec{r}(\xi,\eta)=\sum_{i=1}^{5}\vec{C}_i L_i \tag{3.5}$$

The last one of the basis functions can make the transformation quadratic in ξ, and at the same time it vanishes when $\eta=1$, which makes *Edge 34* in Figure 4 linear in ξ, and thus straight. Here $\vec{r}=(x,y)$ is the position vector in the physical domain.

A total of 10 scalar coefficients ($\vec{C}_i$, $i=1\ldots5$) need to be determined. Since the 5 points ($\vec{P}_i$, $i=1\ldots5$) are already well defined, the coefficients can be uniquely computed. In fact, we obtain the following 5 equations

$$\begin{aligned}
&\vec{C}_1-\vec{C}_3=\vec{C}_{b,0}\\
&\vec{C}_2-\vec{C}_4=\vec{C}_{b,1}\\
&2\vec{C}_5=\vec{C}_{b,2}\\
&\vec{C}_1+\vec{C}_2+\vec{C}_3+\vec{C}_4=\vec{P}_3\\
&\vec{C}_1-\vec{C}_2+\vec{C}_3-\vec{C}_4=\vec{P}_4.
\end{aligned} \tag{3.6}$$

The solution can be found to be

$$\vec{C}_1=\frac{1}{4}\left(\frac{\vec{P}_0+\vec{P}_2}{2}+\vec{P}_1+\vec{P}_3+\vec{P}_4\right)$$

$$\vec{C}_2=\frac{1}{4}\left(-\vec{P}_0+\vec{P}_2+\vec{P}_3-\vec{P}_4\right)$$

$$\begin{aligned}\vec{C}_3 &= \frac{1}{4}\left(-\frac{\vec{P}_0+\vec{P}_2}{2}-\vec{P}_1+\vec{P}_3+\vec{P}_4\right)\\ \vec{C}_4 &= \frac{1}{4}\left(\vec{P}_0-\vec{P}_2+\vec{P}_3-\vec{P}_4\right)\\ \vec{C}_5 &= \frac{\vec{P}_0+\vec{P}_2}{2}-\vec{P}_1.\end{aligned} \tag{3.7}$$

3.3 Geometric Mapping Based on a Cubic Bezier Curve

The formulation of a cubic Bezier curve is

$$\vec{B}(t)=(1-t)^3\vec{P}_0+3(1-t)^2t\vec{P}_1+3(1-t)t^2\vec{P}_2+t^3\vec{P}_3,\quad t\in[0,1]. \tag{3.8}$$

As is shown if Figure 5, the curve starts at $\vec{P}_0$ going toward $\vec{P}_1$ and arrives at $\vec{P}_3$ coming from the direction of $\overrightarrow{P_2P_3}$. Usually, it does not pass through $\vec{P}_1$ or $\vec{P}_2$. These points are only there to provide directional information. The distance between $\vec{P}_0$ and $\vec{P}_1$ determines "how long" the curve moves into direction $\overrightarrow{P_2P_3}$ before turning towards $\vec{P}_3$.

For a cubic Bezier curve, obviously the tangent directions $\vec{t}_1$, $\vec{t}_2$ and the two end points $\vec{P}_0$ and $\vec{P}_3$ are specified. Furthermore an additional constraint forces the center point of the Bezier curve at t = ½ to coincide with the center point of the curve boundary $\vec{P}'$, which is also specified. In order to completely define the cubic Bezier curve, we need to compute $\vec{P}_1$ and $\vec{P}_2$. It is known that the control points $\vec{P}_1$ and $\vec{P}_2$ should be along the tangent directions $\vec{t}_1$ and $\vec{t}_2$ respectively. Assume that $\vec{P}_1'$ and $\vec{P}_2'$ are the projections of $\vec{P}_1$ and $\vec{P}_2$ on the straight line $\overline{P_0P_3}$ as shown in Figure 5, and define $\lambda_1=\frac{\left|\overrightarrow{P_0P_1'}\right|}{\left|\overrightarrow{P_0P_3}\right|},\lambda_2=\frac{\left|\overrightarrow{P_3P_2'}\right|}{\left|\overrightarrow{P_3P_0}\right|}$ as the location parameters of $\vec{P}_1$ and $\vec{P}_2$. Then we have

$$\vec{P}_1=\vec{P}_0+\vec{t}_1\frac{\lambda_1\left|\vec{P}_3-\vec{P}_0\right|^2}{\vec{t}_1\bullet\left(\vec{P}_3-\vec{P}_0\right)},\quad \vec{P}_2=\vec{P}_3-\vec{t}_2\frac{\lambda_2\left|\vec{P}_3-\vec{P}_0\right|^2}{\vec{t}_2\bullet\left(\vec{P}_3-\vec{P}_0\right)}. \tag{3.9}$$

The center point of the Bezier curve is defined using *t=1/2*

$$\begin{aligned}\vec{B}\left(\frac{1}{2}\right)&=\frac{1}{8}\left(\vec{P}_0+3\vec{P}_1+3\vec{P}_2+\vec{P}_3\right)\\&=\frac{\vec{P}_0+\vec{P}_3}{2}+\frac{3l}{8}\left(\frac{\lambda_1}{\cos\theta_1}\vec{t}_1+\frac{\lambda_2}{\cos\theta_2}\vec{t}_2\right)=\vec{P}'\end{aligned}\tag{3.10}$$

where $\cos\theta_1=\frac{\vec{t}_1\bullet\left(\vec{P}_3-\vec{P}_0\right)}{\left|\vec{P}_3-\vec{P}_0\right|},\cos\theta_2=\frac{\vec{t}_2\bullet\left(\vec{P}_3-\vec{P}_0\right)}{\left|\vec{P}_3-\vec{P}_0\right|}$.

For now we assume that $\vec{t}_1$ and $\vec{t}_2$ are linearly independent, and then Eqn. (3.10) can be used to solve for λ_1 and λ_2.

3.4 Possible Numerical Problems with Bezier Curve based mapping and fixes

Though the Bezier mapping offers a feasible way of achieving C^1 continuous curve boundaries, it is not free of numerical problems. For example, all the equations in Section 2-3 assume $\vec{t}_1$ and $\vec{t}_2$ are non-parallel. If in reality they are close to parallel, both P-2 and P-3 Bezier curve may have problems. In the actual solver, these segments can be simply left straight without losing any geometric feature in most cases.

Since sharp corners do exist in certain geometries, such as the trailing edge of an airfoil, it needs to be made sure that these real sharp corners are not smoothed out when average normals are used. Therefore, before the average normal of two adjacent linear segments is computed, the angle between the two original normals is checked, if the angle is larger than a threshold, the node is marked as an end point of the curve boundary and the average normal will not be used. It is found that this treatment is normally quite sufficient to preserve the real sharp corners in geometry.

The P-2 Bezier curve, due to its lower polynomial degree, cannot resolve an inflection point. An inflection point shown in Figure 6a will result in an invalid mapping with negative Jacobians as shown in Figure 6b. In this case, the segment is also left straight, i.e. unchanged.

If the average normal of two adjacent segments is used, the P-2 Bezier curve also has trouble where the curve boundary ends, as indicated in Figure 7, because there is no further tangential information to average from. In this case, with the tangential directions like those in Figure 8a, a singularity would form and invalidate the mapping. The fix to this problem is to extrapolate the normal direction based on a uniform curvature, like in Figure 8b.

The P-3 Bezier curve does not have these symptoms suffered by the P-2 Bezier curve. However, things can still go wrong due to the imperfect geometry information. Problems arise if one or both of λ_1 and λ_2 solved from (3.10) are negative, close to zero or too large. To ensure the robustness, non-dimensional parameters λ_1 and λ_2 are limited to a certain positive range. The range $0.16<\lambda_{1,2}<0.33$ is used in all the test cases.

4. Numerical Tests

4.1 Geometry approximation test with a circular arc segment

For the first test case, a simple geometry is approximated by different types of curves, and thus the relative accuracy of the geometry mappings can be visualized. The tested geometry is an arc from the unit circle, with $\frac{\pi}{2} \le \theta \le \frac{3\pi}{4}$. For the IP mapping, the interior nodes are distributed as Chebyshev-Lobatto points [8] according to the arc length. In Figure 9, the error from the exact geometry is enlarged by 50 times to distinguish different curves. From this figure, for the IP mapping, the geometry error is reduced as the order of the polynomial increases. The P-2 Bezier boundary has much larger error than the other approaches, but the exact slopes at both end points are preserved. The P-5 and P-3 Bezier curves both produce minimal geometry errors and it is impossible to tell the difference here.

In Figure 10, the geometry error is enlarged by 20,000 times to compare the P-5 and P-3 Bezier boundaries. Obviously, the P-5 curve by far produces the smallest geometry error.

As accurate as it is, the P-5 IP mapping is highly unrealistic for many real-world geometries with unknown exact definitions. However, the Bezier curve based boundaries may have some advantage, since they relies less on the interior nodes.

4.2 Subsonic flow around a circular cylinder

A subsonic flow field with Mach=0.3 around a cylinder is computed with different geometry mapping approaches. A structured 16X7 mesh shown in Figure 11 is used in all the simulations. The mesh has a domain size of 20 times the radius. The exact geometry is used for the interpolation nodes and normal directions. For all the cases, the solver is able to converge to machine zero.

Both the IP and Bezier mappings are tested with the 6th order LCP scheme, and the results are shown in Figure 12. The P-3 boundary causes very large entropy error at nodes between elements. The P-5 boundary, however, is able to take full advantage of the 6th order scheme, because the boundary is of the same order of accuracy as the solution. The P-3 Bezier boundary produces almost identical entropy error as the P-5 boundary. The P-2 Bezier does generate large errors at a few points, but it outperforms the P-3 boundary.

The verification of the order of accuracy is shown in Figure 13. Both the P-5 and P-3 Bezier type boundaries are able to achieve optimal convergence, as seen from the test from the 2nd – 6th order cases.

An extremely coarse mesh shown in Figure 14 is used to test the P-5 and P-3 Bezier boundaries. A smooth solution can be achieved for both cases as displayed in the same figure. In terms of the entropy error, the P-3 Bezier boundary performs much better by producing 3 orders smaller error than that by P-5 boundary as shown in Figure 15. This case demonstrated that slope continuity proves to be more essential for coarser meshes.

4.3 Subsonic flow around a NACA 0012 airfoil

NACA 0012 is a more general geometry to test how the mapping approaches work in real-life cases. The computational mesh is a mixed mesh shown in Figure 16. The Mach number is set to 0.3 and the angle of attack is 5^o. The entropy errors on the upper surface of the airfoil computed with different boundary representations are compared.

The exact geometry of the NACA0012 airfoil can be expressed as

$$y = \pm 0.6\left(0.2969\sqrt{x} + 0.1260x - 0.3537x^2 + 0.2803x^3 - 0.1015x^4\right) \tag{4.1}$$

With this geometry information, P-5, P-3 and P-2 mapping can use the exact interior nodes inserted according to arc length. P-2 Bezier mapping can use the exact normal at the end nodes. P-3 Bezier can take advantage of both exact normal and exact interior nodes.

The entropy error on the upper airfoil surface is shown in Figure 17. To show the entropy error near the leading edge more clearly, a polar coordinate system is used, with its origin at (0, 0), i.e. the leading edge. In the figure, 0 degree stands for the trailing edge, while 90 degrees corresponds to the leading edge.

From the figure, all the mapping methods produce large entropy errors near the trailing edge. In the smooth region of the airfoil, the P-5 mapping produces the smallest entropy error, and the P-3 mapping also performs very well. However, when it comes to the region near the leading edge, the P-3 Bezier mapping produces much less entropy error. The P-2 Bezier mapping is able to outperform the P-3 mapping, too.

Figure 18 shows the comparison of the entropy error field by P-5 and P-3 Bezier near the leading edge. It can be further demonstrated that P-5 performs better in smooth regions while P-3 Bezier is superior in areas with large curvatures and less grid resolution.

Another way of the numerical test, however, is assuming we do not know the analytical definition (4.1). This means the only known geometry information is the nodes and linear segments from the mesh.

For the IP mapping, the interior nodes are inserted by cubic reconstruction: for each straight-line segment, four nearest boundary nodes are found, from which a cubic polynomial can be reconstructed. The approximate interior nodes are then taken from the polynomial. This method works up to the P-3 mapping. For the P-5 mapping, a much more complicated degree 5 polynomial reconstruction is needed, and thus it is not included in this test.

For the Bezier mapping, the approximate normal directions are obtained by averaging the normal of adjacent linear segments. For P-3 Bezier, the center node is also inserted from cubic reconstruction.

The entropy distribution on the upper surface is shown in Figure 19. The P-2 and P-3 boundaries perform much worse. The entropy errors are almost 100 times as large as the cases with exact geometry. On the other hand, the P-2 and P-3 Bezier mapping performance suffer much less deterioration. In smooth regions, the P-3 Bezier mapping performs the best, but it loses its advantage over the P-2 Bezier mapping near the leading edge. This is mainly due to the less accurate cubic reconstruction in regions with large curvature.

Based on this test, the Bezier mappings seem to have a significant advantage for general complicated geometries. In most cases, the exact geometry definition is not available.

5. Conclusions

A new type of geometric mapping technique based on Bezier curves is developed and tested with a high-order Euler solver. The new mapping ensures C^1 continuity between adjacent curve segments, and shows a great potential in various numerical tests.

For simple shapes, the new approach can effectively reduce the entropy error with a very coarse mesh. For complicated geometries, where conventional methods rely heavily on the knowledge of exact geometry information, the new approach is able to maintain a low entropy error even without exact geometry information.

Acknowledgement

The study was funded by AFOSR grant FA9550-06-1-0146, and partially by DOE grant DE-FG02-05ER25677. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of AFOSR, DOE, or the U.S. Government.

References

[1] F. Bassi, S. Rebay, High order accurate discontinuous finite element solution of 2D Euler equations, Journal of Computational Physics 138 (1997) 251-285.

[2] F. Bassi, S.Rebay, Accurate 2D computations by means of a high order discontinuous finite element method, in: XIV International Conference on Numerical Methods in Fluid Dynamics, Lecture Notes in Physics, vol. 453, Springer, 1994, pp.234-240.

[3] B. Cockburn, C.-W. Shu, TVB Runge-Kutta local projections discontinuous Galerkin finite element method for conservation laws II: general framework, Math Comput. 52 (1989) 411-435.

[4] B. Cockburn, C.-W. Shu, The Runge-Kutta discontinuous Galerkin method for conservation laws V: multidimensional systems, J. Comput. Phys 141 (1998) 199-224.

[5] J. D. Foley, A. van Dam, S. Feiner, J. Hughes: Computer Graphics: Principles and Practice in C, 2nd ed., Addison Wesley, 1992.

[6] W. J. Gordon and C. A. Hall, "Construction of Curvilinear Co-ordinate Systems and Applications to Mesh Generation," Int. J. Num. Meth. Engng., 7 (1973), pp 461-477.

[7] W. J. Gordon and C. A. Hall, Transfinite element methods and blending function interpolation over curved element domains. Numer. Math. 21(1973), 109-129.

[8] H.T. Hynh, A Flux Reconstruction Approach to High-Order Schemes Including Discontinuous Galerkin Methods, AIAA-2007-4079.

[9] D. A. Kopriva and J. H. Kolias, A conservative staggered grid Chebychev multi-domain method for compressible flows. J. Comput. Phys. 125 (1996), 1, 244–261.

[10] L. Krivodonova, M. Berger, High-order accurate implementation of solid wall boundary conditions in curved geometries, J. Comput. Phys. 211 (2006) 492-512.

[11] Y. Liu, M. Vinokur, and Z.J. Wang, Discontinuous Spectral Difference Method for Conservation Laws on Unstructured Grids, Journal of Computational Physics Vol. 216, pp. 780-801 (2006).

[12] Z.J. Wang, Y. Liu, Spectral (finite) volume method for conservation laws on unstructured grids III: extension to one-dimensional systems, J. Sci. Comput. 20 (2004) 137.

[13] Z.J. Wang, L. Zhang, Y. Liu, Spectral (finite) volume method for conservation laws on unstructured grids III: extension to one-dimensional systems, J. Comput. Phys. 194 (2004) 716.

[14] Z.J. Wang, Y. Liu, Extension of the spectral volume method to high-order boundary represention, J. Comput. Phys. 211 (2006) 154-178.

[15] Z.J. Wang, Y. Liu, G. May and A. Jameson, “Spectral Difference Method for Unstructured Grids II: Extension to the Euler Equations,” Journal of Scientific Computing, Vol. 32, No. 1, pp. 45-71 (2007).

[16] Z.J. Wang, H. Gao, A Unifying Collocation Penalty Formulation for The Euler and Navier- Stokes Equations on Mixed Grids, AIAA-2009-0401 (2009).

[17] O.C. Zienkiewicz, R.L. Taylor, The finite Element Method The Basics, vol.1, Butterworth-Heinemann, Oxford, England, 2000

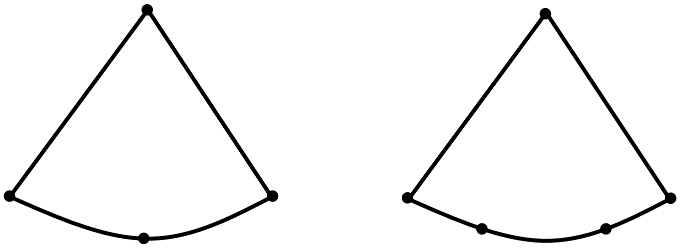


Figure 1. Simplified quadratic and cubic triangular elements with one curve side.

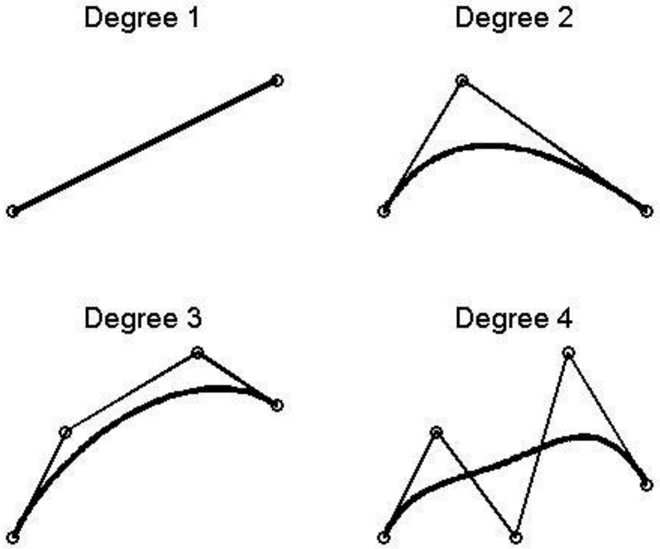


Figure 2. Bezier curves of various degrees.

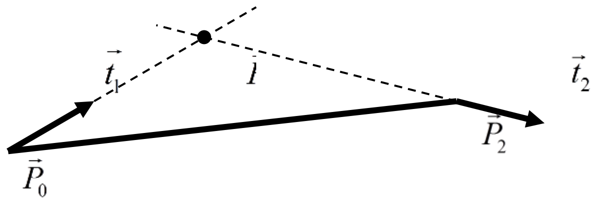


Figure 3. Schematic of quadratic Bezier curve.

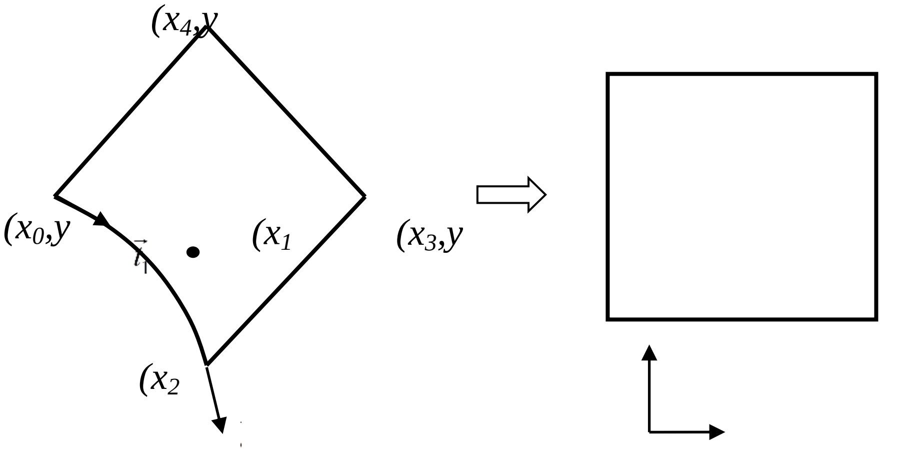


Figure 4. Geometry mapping for a quadrilateral cell with one curved side.

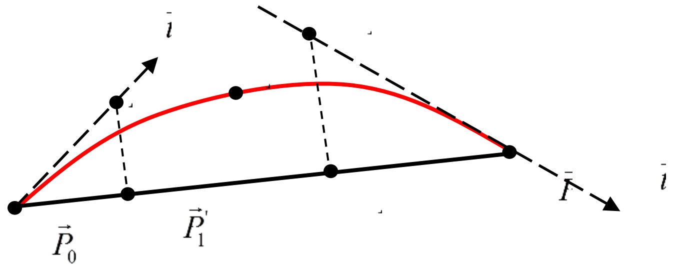
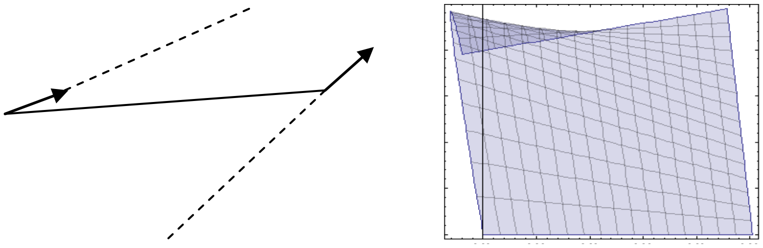


Figure 5. Construction of a cubic Bezier curve boundary segment.



(a) (b)

Figure 6. (a) Tangential directions due to a point of inflection and (b) resulted transformation.

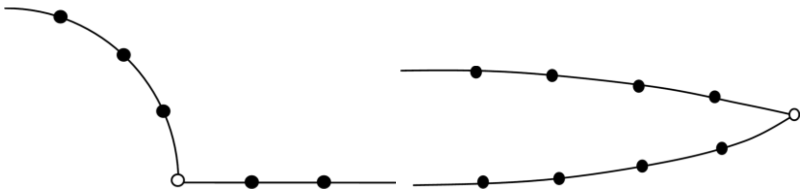
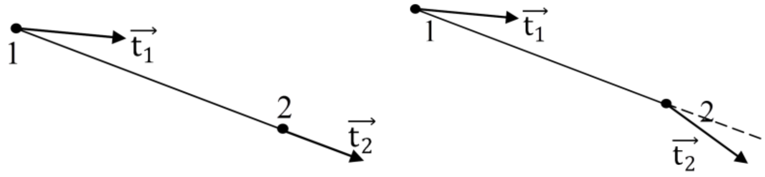


Figure 7. Problematic nodes (in white) for P-2 Bezier curves.



(a) (b)

Figure 8. (a) The tangential directions where the curve boundary ends

(b) The fix to the singularity problem by extrapolating $\vec{t}_2$ with $\angle 1 = \angle 2$.

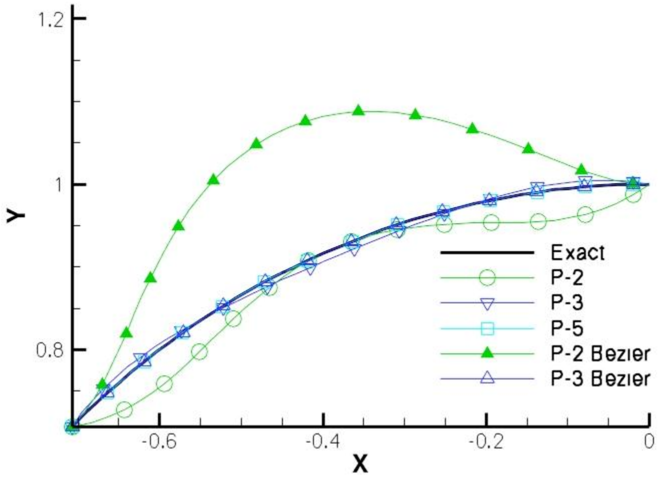


Figure 9. Approximations of a circular segment by different approaches, the errors enlarged by 50 times.

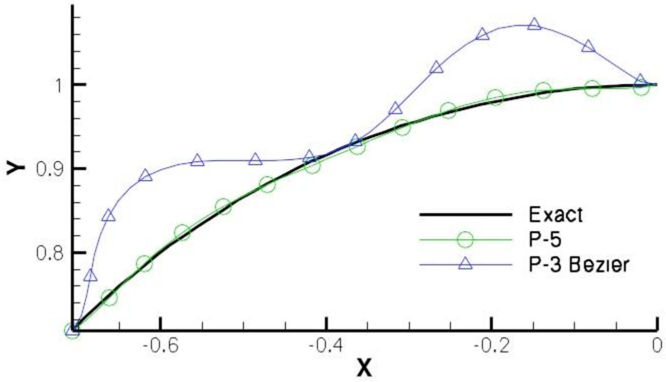


Figure 10. Comparison of the P-5 and P-3 Bezier representations, with errors enlarged by 20,000 times.

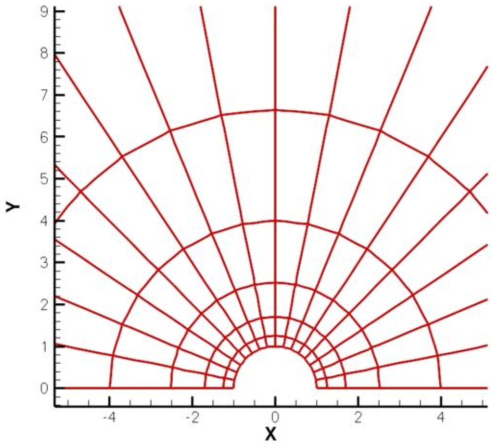


Figure 11. Computational mesh used for flow over a cylinder.

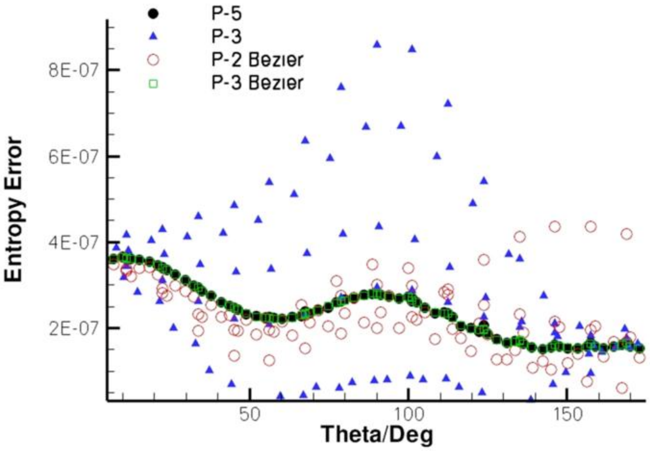


Figure 12. Entropy error distributions on the cylinder wall, 6th order LCP.

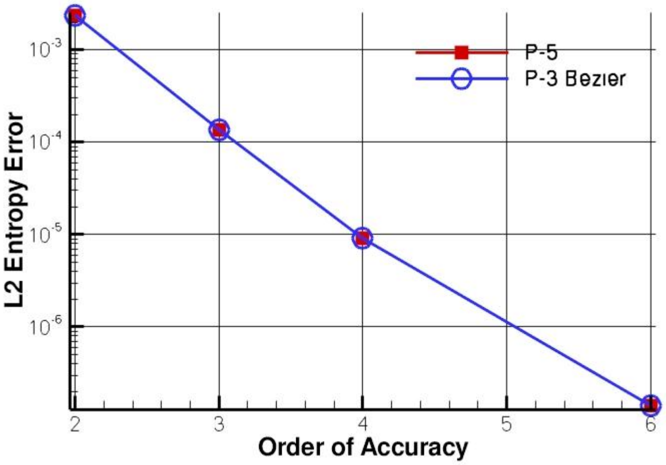


Figure 13. L_2 entropy errors computed with 2nd – 6th order schemes.

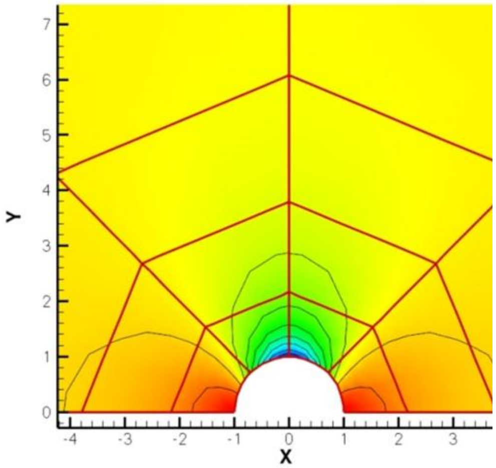


Figure 14. A 4X6 mesh around the cylinder.

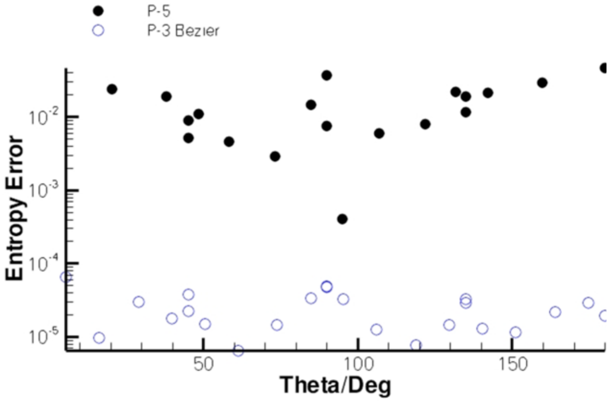


Figure 15. Entropy error distributions on a coarse mesh for the flow over a cylinder.

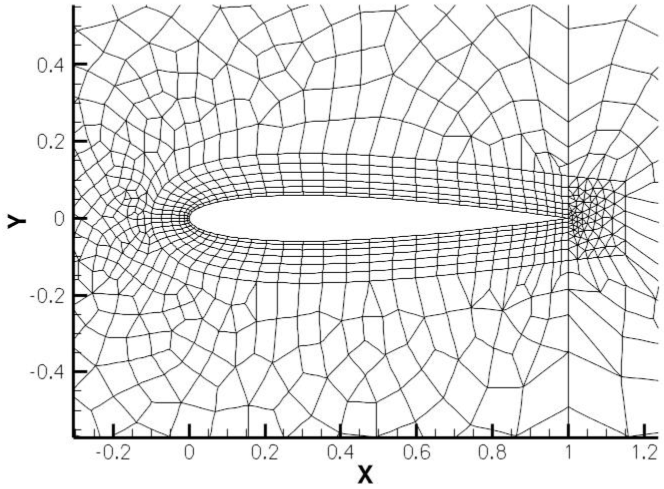


Figure 16. A mixed mesh for NACA0012 test case.

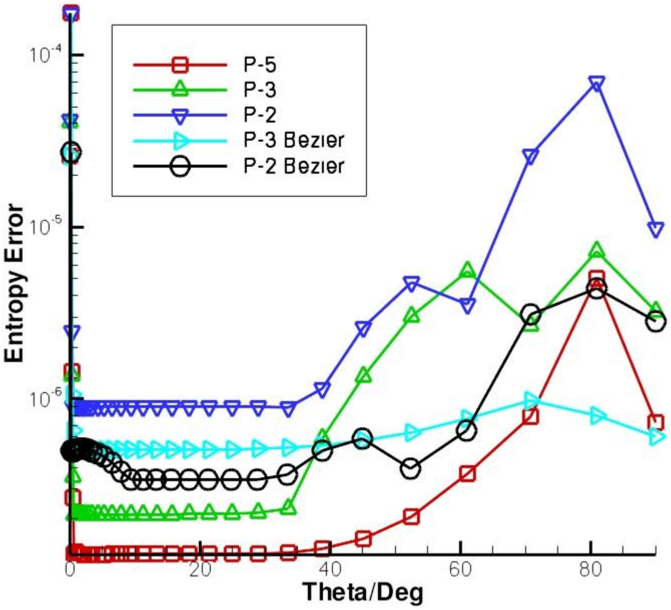
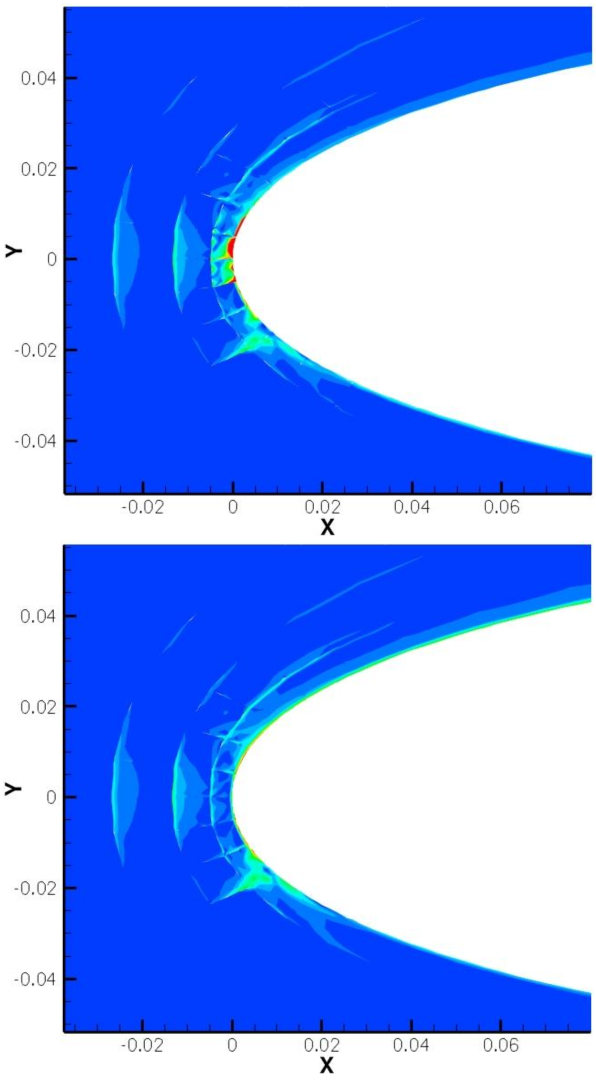


Figure 17. Entropy errors on the upper wall of NACA0012 airfoil with a 6th order scheme, computed with exact geometry information.



a) P-5

b) P-3 Bezier

Figure 18. Entropy error field near the leading edge of NACA0012 airfoil with the 6th order scheme.

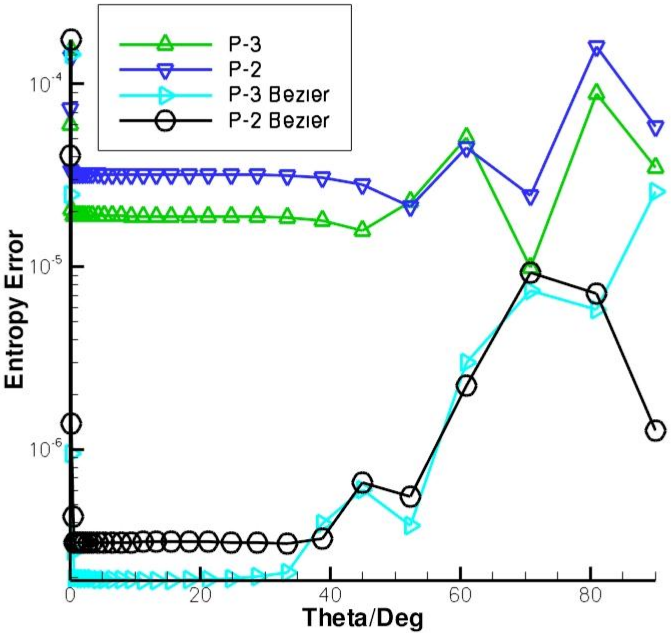


Figure 19. Entropy errors on the upper wall of NACA0012 airfoil with a 6th order scheme, computed without exact geometry information.

CHAPTER 5. A Residual-Based Procedure for hp-Adaptation on 2D Hybrid Meshes

A paper published in 49th AIAA ASM conference proceedings

Haiyang Gao and Z.J. Wang

Abstract

H- and p-adaptations are effective ways of achieving high-quality solutions for CFD simulations, with reasonable computing costs. Combined hp-adptation offers even greater flexibility for complex problems. In the present study, a residual based procedure for hp-adaptaion is developed. The adaptation is driven by an efficient multi-p a posteriori error estimator. P-adaptation is applied to smooth regions of the flow field while h-adaptation targets the non-smooth regions, identified by accuracy-preserving TVD marker. Several numerical tests with the correction procedure via reconstruction (CPR) formulation are presented to demonstrate the capability of the technique.

1. Introduction

Adaptive solution methods have been an active topic of CFD research for decades. Through these methods, computing power can be redistributed to where it is needed the most, thus the desired accuracy can be achieved with minimal costs.

For a high-order method, two types of adaptations are usually considered: local mesh subdivision (h-refinement) and local polynomial degree enrichment (p-refinement). It has been shown that p-adaptation is superior in regions where the solution is smooth [13], while h-adaptation is preferred in non-smooth regions such as flow discontinuities, where high-order methods become less effective. Obviously, the combination of both h- and p-adaptation offers greater flexibility than methods that incorporates h- or p-adaptation alone.

Hp-adaptation is made possible by the development of a series of compact high-order methods for unstructured meshes, such as the discontinuous Galerkin (DG) method [3,4,7], the spectral volume (SV) method [14,21] and the spectral difference (SD) method [15]. Unlike the finite volume method that achieve high-order by expanding the stencil, the above methods employ a polynomial to approximate the solution in each cell or element, and the polynomials collectively form a function which is discontinuous across cell boundaries. The compact solution reconstruction of these methods is an ideal feature for p-adaptation. Recently, the correction procedure via reconstruction (CPR) formulation was developed in 1D [11], and extended to simplex and hybrid meshes in [22]. It unites the DG, SV and SD method into a nodal differential formulation. The CPR formulation combines the compactness and high accuracy with the simplicity and efficiency of the finite difference method, and can be easily implemented for arbitrary unstructured meshes. The current hp-adaptation is based on the CPR formulation, but similar methods can be easily applied to other campact high-order methods.

For an hp-adaptation scheme, two indicators needs to be computed: a local error indicator to serve as the adaptation criteria, and a local smoothness indicator, used to decide whether h- or p-adaptation should be applied to a certain region.

The error indicator is the foundation of any adaptation method, and finding a reliable error indicator is no trivial task. It seems there are at least three types of adaptation criteria: gradient or feature based error estimate [5,9], residual-based error estimate [1,2,6,17,24], and adjoint-based error estimate [8,10,16,18-20]. The first criteria are the most widely used: it is normally quite easy to implement, since the error indicator is computed directly from solution variables. But the approach is also the most ad-hoc, for it often involves case-dependent parameters. When properly used, it can improve solution quality efficiently [9]. The residual-based error estimate has a variety of forms. It can be used with a multi-grid approach [2], or with an approximate error transport equation [24]. There are also residual estimates based on the solution discontinuity across cell boundaries [17]. For “real world” 3D problems, the multi-grid approach requires the generation of fine grids, which may be difficult to generate. This has probably hindered its wider application. The most successful approach is arguably the adjoint-based approach, which relates a specific functional output, such as the lift or drag coefficient in aerodynamics, directly to the local residual by solving an additional adjoint problem. It has been shown that this approach is very effective in driving an hp-adaptation procedure in obtaining a very accurate prediction of the functional output [18]. The drawback of this approach is that an adjoint problem must be solved, and a functional output is required to drive the adaptation.

In the present study, a residual-based approach using a multi-p error estimation is adopted as the error indicator. This error indicator involves no ad-hoc parameters. It is also much more efficient to compute compared to the adjoint-based approach, since it mostly involves residual evaluation.

For the search of the smoothness indicator, we turn to the solution limiting schemes for high-order methods. The “troubled cell” markers are often used to mark the non-smooth cells before the solution is limited. The accuracy-preserving TVD (AP-TVD) marker [23] is employed in the present study as the non-smoothness indicator, since it does not have case-dependent parameters and can effectively mark the non-smooth regions without marking the smooth extremas. For non-smooth regions, a hierarchical moment limiter [23] is applied to suppress the spurious oscillations.

The rest of the paper is organized as follows: the numerical formulation is presented in Section 2, including a brief review of the CPR formulation, the multi-p error estimation, and the AP-TVD marker. Section 3 is about mesh adaptation strategies, including the treatment of non-conformal interfaces in hp-adaptation and the procedures for hp-adaptation. Several numerical test cases of the inviscid flow are presented in Section 4. Section 5 is devoted to concluding remarks.

2. Numerical Formulation

Review of the CPR formulation

Consider the hyperbolic conservation law,

$$\frac{\partial Q}{\partial t}+\vec{\nabla}\bullet\vec{F}(Q)=0, \qquad (2.1)$$

where $\vec{F}=(F,G)$ is the flux vector. The computational domain is discretized into N non-overlapping elements $\{V_i\}$. Let W be an arbitrary weighting function. The weighted residual of Eq. (2.1) on element V_i can be written as

$$\int_{V_i}\left(\frac{\partial Q}{\partial t}+\vec{\nabla}\bullet\vec{F}(Q)\right)WdV=\int_{V_i}\frac{\partial Q}{\partial t}WdV+\int_{\partial V_i}W\vec{F}(Q)\bullet\vec{n}\,dS-\int_{V_i}\vec{\nabla}W\bullet\vec{F}(Q)dV=0. \qquad (2.2)$$

Let Q_i be an approximate solution to Q at element i. We assume that in each element, the solution belongs to the space of polynomials of degree k or less, i.e., $Q_i\in P^k$, with no continuity requirement across element interfaces. The numerical solution Q_i should satisfy Eq. (2.2), i.e.,

$$\int_{V_i} \frac{\partial Q_i}{\partial t} W dV + \int_{\partial V_i} W \vec{F}(Q_i) \bullet \vec{n} dS - \int_{V_i} \vec{\nabla} W \bullet \vec{F}(Q_i) dV = 0. \tag{2.3}$$

A common Riemann flux across cell interfaces is used to replace the normal flux to provide element coupling, i.e.,

$$\vec{F}(Q_i) \bullet \vec{n} \approx F_{\text{com}}^{n}(Q_i, Q_{i+}, \vec{n}) \cdot \tag{2.4}$$

where Q_{i+} is the solution on V_{i+}, which is outside V_i. Instead of (2.3), the solution is required to satisfy

$$\int_{V_i} \frac{\partial Q_i}{\partial t} W dV + \int_{\partial V_i} W F_{\text{com}}^{n} dS - \int_{V_i} \vec{\nabla} W \cdot \vec{F}(Q_i) dV = 0. \tag{2.5}$$

Applying integration by parts to the last term on the LHS of (2.4), we obtain

$$\int_{V_i} \frac{\partial Q_i}{\partial t} W dV + \int_{V_i} W \vec{\nabla} \bullet \vec{F}(Q_i) dV + \int_{\partial V_i} W \left[F_{\text{com}}^{n} - F^{n}(Q_i) \right] dS = 0. \tag{2.6}$$

The last term on the left side of (2.5) can be viewed as a penalty term, i.e., penalizing the normal flux differences. Introduce a "correction field", $\delta_i \in P^k$, which is determined from a "lifting operator",

$$\int_{V_i} W \delta_i dV = \int_{\partial V_i} W [F^n] dS, \tag{2.7}$$

where $[F^n] = F_{\text{com}}^{n}(Q \;\; , Q^+, \vec{n}) - F^n(Q \;\;)$ is the normal flux difference. Substituting (2.7) into (2.6), we obtain,

$$\int_{V_i} \left[\frac{\partial Q_i}{\partial t} + \vec{\nabla} \bullet \vec{F}(Q_i) + \delta_i \right] W dV = 0. \tag{2.8}$$

For non-linear fluxes, the flux divergent $\vec{\nabla} \bullet \vec{F}(Q_i)$ is usually not in the space P^k, in which case $\vec{\nabla} \bullet \vec{F}(Q_i)$ is projected to P^k by,

$$\int_{V_i}\Pi\left(\vec{\nabla}\cdot\vec{F}\left(Q_i\right)\right)W\mathrm{d}V=\int_{V_i}\vec{\nabla}\cdot\vec{F}\left(Q_i\right)W\mathrm{d}V \tag{2.9}$$

Because W is arbitrary, (2.8) is equivalent to

$$\frac{\partial Q_i}{\partial t}+\Pi\left(\vec{\nabla}\cdot\vec{F}\left(Q_i\right)\right)+\delta_i=0\,. \tag{2.10}$$

With the definition of a correction field δ_i, we have successfully reduced the weighted residual formulation to an equivalent simple differential form, which does not involve any explicit surface or volume integrals. The lifting operator obviously depends on the choice of weighting function. If $W\in P^k$, (2.10) is equivalent to the DG formulation. The performance of this formulation of course hinges on how efficiently the correction field δ_i can be computed. If $[F^n]$ is assumed to be degree p polynomials along the cell faces, and that the triangle has straight faces, the correction field δ_i can be computed explicitly in the following form

$$\delta_{i,j}=\frac{1}{\left|V_i\right|}\sum_{f\in\partial V_i}\sum_{l}\alpha_{j,f,l}[F^n]_{f,l}S_f\,, \tag{2.11}$$

where $\alpha_{j,f,l}$ are constant lifting coefficients independent of the solution, but dependent on the weighting function W. Substituting (2.11) into (2.10) we obtain the following formulation

$$\frac{\partial Q_{i,j}}{\partial t}+\Pi\left(\vec{\nabla}\cdot\vec{F}\left(Q_i\right)\right)+\frac{1}{\left|V_i\right|}\sum_{f\in\partial V_i}\sum_{l}\alpha_{j,f,l}[F^n]_{f,l}S_f=0\,. \tag{2.12}$$

With a judicious choice of solution and flux points, as shown in Figure 1, solution reconstruction can be avoided. Other advantages include an identity mass matrix, and easy extension to mixed elements such as prismatic, and pyramidal cells in 3D. For curved iso-parametric elements, the CPR formulation is applied to the transformed equation on a standard element. The application of the CPR formulation to hybrid meshes is quite straight forward. It has been successfully applied to arbitrary 2D hybrid meshes.

Multi-p residual-based error indicator

To present the basic idea, we assume that the solution is sufficiently smooth, and p-refinement is chosen. Let the (nonlinear) differential operator be L, and the differential equation be denoted by $L(Q) = 0$ with Q the exact solution. Denote L^{Hp} the discrete CPR operator on a mesh of size H with a given polynomial degree p, and Q^{Hp} the numerical solution. Obviously, we have

$$L^{Hp}\left(Q^{Hp}\right) = 0. \tag{2.13}$$

Let's assume the numerical solution is reasonably close to the exact solution. The solution error should then be $(p+1)$th order.

Then, we could interpolate or prolongate the lower order (degree p) solution to a higher order P (e.g. $P = p+1$) solution space. The residual of the higher order operator on the prolongated solution can serve as an error estimate. Let the prolongated solution be Q^{HP}. Denote this interpolation operator I_p^P, then

$$Q^{HP} = I_p^P Q^{Hp} \tag{2.14}$$

The defect is defined as $d^{HP} = I_p^P\left[L^{Hp}\left(Q^{Hp}\right)\right] - L^{HP}\left(I_p^P Q^{Hp}\right)$, where L^{HP} is the discrete higher order operator. Ideally, the first term $I_p^P\left[L^{Hp}\left(Q^{Hp}\right)\right]$ should be 0, but in real applications, the solution does not need to converge completely in order to calculate d^{HP}. Experiences with multigrid-based error estimators indicate the defect d^{HP} is an excellent indicator of the error sources, which generate solution errors to appear possibly at other regions because of the transport nature of the governing equations.

Then, the defect is localized to each cell by

$$\eta_i = \frac{1}{V_i}\int_{V_i} \left| d_i^{HP} \right| dV . \tag{2.15}$$

The quantity η_i is then used as the error indicator to drive the hp-adaptation. Note that the more expensive high-order residual operator is only used once during the evaluation of η_i, without the need to solve any non-linear high-order problem. Therefore, this operation requires even less computing power than an explicit time marching.

Once the cellwise residual error indicator η is computed for each cell in the domain by Eqn. (2.15), a fixed fraction approach is employed to determine the cells adaptations will be applied to. This approach flags a fixed portion (e.g. $f_{adapt} = 0.1$) of the total cells with the largest error indicator η. Whether h- or p-adaptation is applied will be decided by the AP-TVD marker.

Accuracy-preserving TVD marker

The AP-TVD limiter is used as a smoothness indicator to flag the non-smooth regions of the flow field, such as flow discontinuities. In the flagged regions, h-adaptation is the preferred approach. Once a cell is flagged by the AP-TVD marker, a

The procedure of the AP-TVD marker is divided into two steps:

1. Compute the cell averaged solutions at each cell. Then compute the min and max cell averages for cell *I* from a local stencil including all of its node neighbors, i.e.,

$$\bar{Q}_{i,\min} = \min\left(\bar{Q}_i, \min_{1\le r\le nb} \bar{Q}_r\right), \bar{Q}_{i,\max} = \max\left(\bar{Q}_i, \max_{1\le r\le nb} \bar{Q}_r\right) \qquad (2.16)$$

2. Cell *i* is considered a candidate for trouble cell if, for any solution point *j*,

$$Q_{i,j} > 1.01\bar{Q}_{i,\max} \text{ or}$$

$$Q_{i,j} < 0.99\bar{Q}_{i,\min} \qquad (2.17)$$

The constants 1.01 and 0.99 are not problem dependent. They are simply used to overcome machine error, when comparing two real numbers, and to avoid the trivial case of when the solution is constant in the region.

3. This step aims to unmark those cells at local extrema that are unnecessarily marked in the last step. Here, for a given marked cell, a minmod TVD function is applied to verify whether the cell-averaged 2nd derivative is bounded by the slopes computed by cell-averaged 1st derivatives.

Assume cell *i* is a marked cell in step 2, cell *nb* is a face neighbor of cell *i*, the unit vector in the direction connecting the cell centroids of cell *i* and *nb* is $\vec{l} = (l_x, l_y)$. The cell averaged 2nd derivative in this direction is

$$\bar{Q}_{i,ll} = \bar{Q}_{i,xx} l_x^2 + 2\bar{Q}_{i,xy} l_x l_y + \bar{Q}_{i,yy} l_y^2 \tag{2.18}$$

In a similar manner, the cell-averaged first derivative in *l* direction, for both *i*-th and *nb*-th cells, can be computed as

$$\bar{Q}_{i,l} = \bar{Q}_{i,x} l_x + \bar{Q}_{i,y} l_y \qquad \bar{Q}_{nb,l} = \bar{Q}_{nb,x} l_x + \bar{Q}_{nb,y} l_y \tag{2.19}$$

Then, the estimation of the 2nd derivative can be obtained as

$$\tilde{Q}_{i,ll} = \frac{\bar{Q}_{nb,l} - \bar{Q}_{i,l}}{|\vec{r}_i - \vec{r}_{nb}|} \tag{2.20}$$

Then define a scalar limiter for this face as

$$\phi_{i,nb}^{(2)} = \min\text{mod}\left(1, \frac{\beta \tilde{Q}_{i,ll}}{\bar{Q}_{i,ll}}\right), \tag{2.21}$$

where *β=1.5*. This process is repeated for every face neighbor, and minimum of all computed $\phi_{i,nb}^{(2)}$ is obtained

$$\phi_i^{(2)} = \min_{nb} \phi_{i,nb}^{(2)} \tag{2.22}$$

if $\phi_i^{(2)} = 1$, then cell *i* is unmarked as a non-smooth cell. If $\phi_i^{(2)} < 1$, the cell is confirmed as a trouble cell, and flagged for h-adaptation.

The above non-smoothness marker can be applied only when polynomial degree *p>1* (3rd order and up) for cell *i*. For those cell with *p=1*, a similar procedure involving the cell-averaged 1st derivatives is used. However, the marker for *p=1* is unable to unmark the smooth extremas as the above limiter. Therefore, for cases that have both h- and p-adaptation, the base solution is always computed with 3rd order scheme.

3. Mesh Adaptation

Non-conformal interfaces in hp-adaptation

Changing the h- and p-levels for cells will inevitably create non-conformal interfaces between cells. When a cell is h-adapted, the cell is divided into four new ones by connecting the center of each face, for both triangular and quadrilateral cases. A hanging node is resulted in the adapted mesh between different h-levels. For p-adaptation, the interface between cells with different p-level is also non-conformal, since the flux points on opposing sides support polynomials of different degrees. Special treatment is required when computing the common numerical flux on these non-conformal interfaces.

In the current study, a set of rules are observed when flagging the cells for adaptation. They help achieve a smooth adaptation, also limit the type of interfaces to be dealt with. The rules are listed as follows:

1. The difference of p-levels between neighboring cells are no larger than 1.

2. The difference of h-levels between neighboring cells are no larger than 1, i.e. only one level of hanging nodes are allowed.

3. If one interface has different p-levels on both sides, the h-levels have to be the same.

4. If one interface has different h-levels on both sides, the p-levels have to be the same.

Each time the mesh is adapted, necessary adjustments are made to ensure that the rules above are observed over the whole domain.

When the rules above are enforced, there leaves only three types of interfaces: the normal conformal interfaces, the p-interfaces with the same h-level, and the h-interfaces with the same p-level.

For a non-conformal interface, an intermediate “mortar face” [13] (Figure 2b) is constructed to link the two faces in different solution space. The mortar faces always take the space of the higher p- or h-level of the two sides. First, the solutions from the lower p- or h-level are interpolated or prolongated to the mortar face level. Then, the calculation of common numerical fluxes F_{com}^{n} takes place on the mortar face. Last, the resulted common fluxes need to be projected back to the lower p- or h-space.

The interpolation process is quite straightforward. For both types of interfaces we are dealing with, the lower p- or h-space is always included in the higher p-or h-space, so the same polynomial are used to obtain more point values that match the flux points on the other side. For the restriction on the last step, we use the following

$$\int_f W_j F_n^{h,P} \mathrm{d}S = \int_f W_j F_n^{h,p} \mathrm{d}S \,, \tag{3.1a}$$

$$\int_f W_j F_n^{h,p} \mathrm{d}S = \int_f W_j F_n^{H,p} \mathrm{d}S \,, \tag{3.1b}$$

where W_j *(j=1…p+1)* are the basis of *p*-th degree polynomial. Note that when *W=1*, the conservation condition is automatically satisfied. Eqn (3.1a) is used for p-interface while (3.1b) for h-interface.

One example of this procedure is shown in Figure 2, which is a p-interface with *p-1* on one side and *p-2* on the other (Figure 2a). The *p=1* solution is interpolated to *p=2* (Figure 2b), and the Riemann flux is computed on the mortar face. After that, the resulting F_{com}^n is projected to p-1 space using (3.1a) for use in the p-1 cell (Figure 2c). A similar procedure is followed for h-interfaces.

The adaptation procedure

The procedure for p-adaptation for steady flow is listed as follows:

1. Solve the flow field until a convergence criterion is reached.
2. Follow the procedure in Section 2.2, compute η_i for each cell.
3. Flag a fixed fraction of all the cells with largest error indicator.
4. Make necessary adjustments to enforce the rules in Section 3.1.
5. Update the mesh, metric terms and connectivities. Inject the solution to new p-levels.
6. Repeat step 1-5.

For hp-adaptation, the starting solution is always computed with uniform *p=2*, in order to use the AP-TVD limiter to correctly identify the non-smooth regions. Once the non-smooth region is flagged, the order of accuracy is dropped to *p=1*. The procedure of hp-adaptation is described as follows.

1. Start the computation with uniform *p=2* over the whole domain.
2. Solve the flow field until residual stops changing.
3. Flag all the cells that have been marked in the last five iterations for h-adaption.
4. Follow the procedure in Section 2.2, compute η_i for each cell, seclect a fixed fraction of cells with the largest η_i.

5. Search the neighbors of the cells flagged in step 3, if a neighboring cell has a large η_i, it is also flagged for h-adaptation. Also flag all the h-adapted cells for p-coarsening, reduce their order of accuracy to $p=1$.

6. Apply p-adaptation to smooth regions.

7. Update the mesh, metric terms and connectivities. Inject the solution to new h/p-levels.

8. Repeat steps 2-7.

The adaptation can keep going until desired number of levels is reached, or an output of interest stops changing.

4. Numerical Tests

P-adaptation for a subsonic flow over a Gaussian bump

The flow over a Gaussian bump is computed with the computational domain shown in Figure 3. The incoming flow has a Mach number of 0.5. Wall boundary is applied for the bump (bottom boundary), characteristic boundary for both inlet and outlet, and symmetry for the top boundary.

The solver starts from uniform 2nd order scheme, and then undergoes six loops of p-adaptation. Adaptation fractions $f_{\text{adapt}} = 0.1$ and $f_{\text{adapt}} = 0.2$ are used.

Figure 4 shows the final adaptation results of the $f_{\text{adapt}} = 0.2$ case. It can be seen the residual-based indicator clearly identify the error source (the bump in this case). The closer to the bump, the adaptation has been applied.

Figure 5 shows the Mach contour of the final solution of the $f_{\text{adapt}} = 0.2$ case. Although the cells have been adapted into 6 different p-levels, the solution is kept rather smooth by the mortar face configuration at p-interfaces.

L_2 entropy error over the whole domain is used as a measure of accuracy, defined by

$$\left\|\Delta_{ent}\right\|_{L_2} = \sqrt{\frac{\int\limits_{\Omega}\left(\dfrac{p/\rho^{\gamma} - p_{\infty}/\rho_{\infty}^{\gamma}}{p_{\infty}/\rho_{\infty}^{\gamma}}\right)^2 dV}{\int\limits_{\Omega} dV}} \,. \tag{4.1}$$

The entropy error of the p-adaptation cases are compared with uniform p-refinement in Figure 6. Entropy error is effectively reduced by the p-adaptation. The same level of error is achieved with much fewer DOFs than the uniform p-refinement.

P-adaptation for a subsonic flow over the NACA0012 airfoil

Next, the p-adaptation is tested in the subsonic flow over the NACA 0012 airfoil, with *Ma=0.3* and AOA=5deg. The hybrid mesh used for this case (as shown in Figure 7a) is constructed with an arbitrary mix of quadrilateral and triangular cells. The adaptation starts from a very crude 2nd order solution (Figure 7b), undergoes five levels of p-adaptation, also with $f_{\text{adapt}} = 0.1$ and $f_{\text{adapt}} = 0.2$.

Figure 8 shows the result of the p-adaptation. The region near the airfoil surface is repeatedly adapted to a very high p-level. The region near the leading edge is also identified as a main source of the residual error. The Mach contour of the final solution is shown in Figure 9.

Figures 10a-b shows the L_2 entropy error and the error of the drag coefficient respectively. For both errors, the p-adaptation scheme demonstrates more efficient error reduction with less DOFs. Note that the entropy error stops converging after the solution reach a certain p-level, due to the large entropy production near the trailing edge.

Hp-adaptation for a transonic flow over the NACA0012 airfoil

The computation of a transonic flow over the same airfoil is used to test the hp-adaptation scheme. For this flow, *Ma=0.8* and AOA=1.25 deg. Shockwaves are expected on both upper and lower surface of the airfoil, where h-adaptation should be applied, while p-adaptation is applied to smooth parts of the flow field.

The base mesh is shown in Figure 11. First, a 3rd order base solution is computed, and the AP-TVD limiter is used to identify the shockwaves and other unsmooth regions. Once flagged as non-smooth, the region is reduced to 2nd order accuracy and no p-adaptation is allowed.

Figure 12 shows the hp-adaptation results after 3 levels of adaptation. We can see that the shockwave regions are correctly detected and repetitive h-adaptation is applied in this region. The region near the trailing edge is also h-adapted. P-adaptation is applied to the non-flagged regions, as cells with higher p-levels are found near the leading edge.

The density contours of the final solution are shown in Figure 13. With much finer mesh due to the h-adaptation, the shockwave is well resolved, while smooth solution is obtained in the p-adapted regions, with minor oscillation behind the shockwave.

5. Conclusion

A residual-based error indicator is developed to drive an hp-adaptation procedure in the current study. The AP-TVD troubled cell marked serves as the smoothness indicator to decide whether h- or p- adaptation should be applied to a certain region. The residual-based indicator requires much less computing cost compared with the more sophisticated adjoint-based error indicators. It is also more rigorous than the gradient or feature based approaches.

Several typicial numerical tests of inviscid flow have been conducted. For smooth flow fields, the p-adaptation driven by the residual-based indicator effectively targets the error source. The adaptation procedure is shown to improve the numerical solution visually, as well as achieve the same error level with fewer DOFs. Hp-adaptation is also tested on a transonic flow with shockwaves. The non-smooth regions of the flow field is correctly detected by the AP-TVD marker, where h-adaptation is applied, while p-adaptation is applied to the smooth regions. High-quality solution can be achieved with relatively low cost after the adaptation. These test cases shows the potential of the residual-based adaptation procedure in complex CFD problems.

Acknowledgments

The present study was funded by AFOSR grant FA9550-06-1-0146. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of AFOSR or the U.S. Government.

References

[1] M. Ainsworth, J.T. Oden, A unified approach to a posteriori error estimation using element residual methods, Numer. Math. 65 (1993) 23–50.

[2] T.J. Baker, Mesh adaptation strategies for problems in fluid dynamics. Finite Elem Anal Des 1997;25:243–73.

[3] F. Bassi and S. Rebay, High-order accurate discontinuous finite element solution of the 2D Euler equations, J. Comput. Phys. 138, 251-285 (1997)

[4] C.E. Baumann, T.J. Oden, A discontinuous hp finite element method for the Euler and Navier–Stokes equations. Int J Num Meth Fluids 1999;31(1):79–95.

[5] M. J. Berger and P. Collela, Local adaptive mesh refinement for shock hydrodynamics, J. Comput. Phys., vol. 82, pp. 64-84, 1989.

[6] R. Biswas, K.D. Devine, and J. Flaherty, Parallel, adaptive finite element methods for conservation laws, Appl. Numer. Math., 14, 1994, pp. 255–283.

[7] B. Cockburn and C.-W. Shu, The Runge-Kutta discontinuous Garlerkin method for conservation laws V: multidimensional systems, J. Comput. Phys., 141, 199 - 224, (1998).

[8] M. B. Giles and N. A. Pierce. Adjoint equations in CFD: Duality, boundary conditions and solution behavior. AIAA paper 97-1850, 1997.

[9] R. Harris and Z.J. Wang, “High-order adaptive quadrature-free spectral volume method on unstructured grids,” Computers & Fluids 38 (2009) 2006–2025.

[10] R. Hartmann and P. Houston. Adaptive discontinuous Galerkin finite element methods for the compressible Euler equations. J. Comput. Phys., 183:508–532, 2002.

[11] H.T. Huynh, A flux reconstruction approach to high-order schemes including discontinuous Galerkin methods, AIAA Paper 2007-4079.

[12] H. Jasak, A.D. Gosman, Residual error estimate for the finite-volume method, Numerical Heat Transfer, 39:1-9, 2001.

[13] Y. Li, S. Premansuthan and A. Jameson, Comparison of h- and p-adaptations for spectral difference method, AIAA 2010-4435.

[14] Y. Liu, M. Vinokur and Z.J. Wang, Three-Dimensional High-Order Spectral Finite Volume Method for Unstructured Grids, AIAA-2003-3837.

[15] Y. Liu, M. Vinokur, and Z.J. Wang, Discontinuous Spectral Difference Method For Conservation Laws On Unstructured Grids, Journal Of Computational Physics Vol. 216, pp. 780-801 (2006).

[16] M. A. Park. Adjoint–based, three–dimensional error prediction and grid adaptation. AIAA paper 2002-3286, 2002.

[17] Shih, T.I-P. and Qin, Y., A Posteriori Method for Estimating and Correcting Grid-Induced Errors in CFD Solutions – Part 1: Theory and Method, AIAA Paper 2007-100.

[18] L. Wang and D. Mavriplis, Adjoint-based h-p Adaptive Discontinuous Galerkin Methods for the Compressible Euler Equations, AIAA-2009-0952.

[19] D.A. Venditti, D.L. Darmofal, Adjoint error estimation and grid adaptation for functional outputs: Application to quasi-one-dimensional flow. J Comput. Phys. 2000;164:204–27.

[20] D.A. Venditti, D.L. Darmofal, Anisotropic grid adaptation for functional outputs: application to two-dimensional viscous flows, J. of Comp. Phys, 187 (2003) 22-46.

[21] Z.J. Wang, Spectral (Finite) volume method for conservation laws on unstructured grids: basic formulation, J. Comput. Phys. Vol. 178, pp. 210-251, 2002.

[22] Z.J. Wang and H. Gao, A Unifying Lifting Collocation Penalty Formulation for the Euler Equations on Mixed Grids, AIAA-2009-401.

[23] M. Yang and Z.J. Wang, A parameter-free generalized moment limiter for high-order methods on unstructured grids, AIAA-2009-605.

[24] X.D. Zhang, J.-Y. Trepanier, R. Camarero, A posteriori error estimation for fnite-volume solutions of hyperbolic conservation laws, Comput. Methods Appl. Mech. Engrg. 185 (2000).

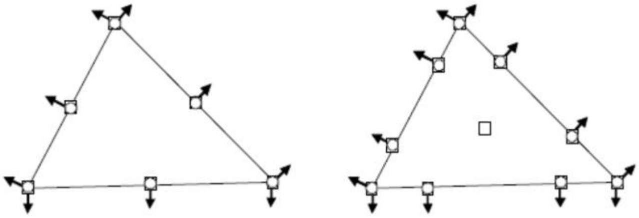
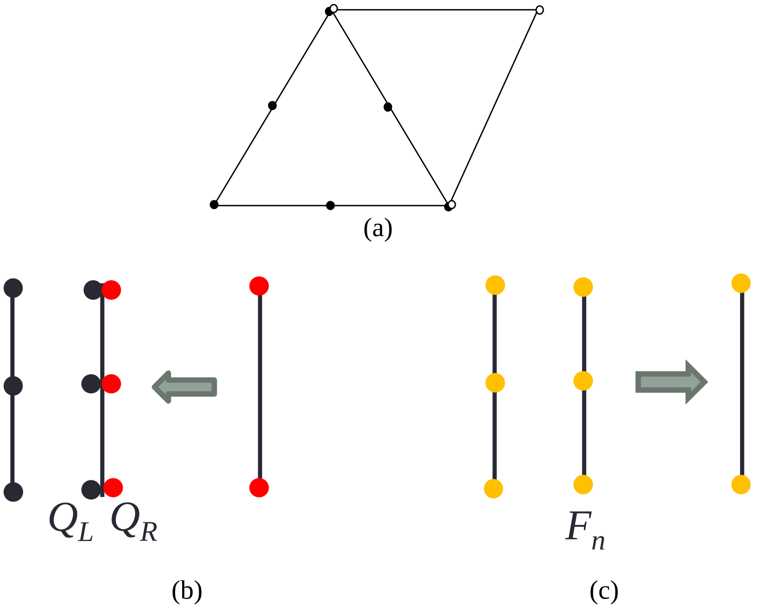


Figure 1. The configuration of solution points (square) and flux points (circle) of p-2 and p-3 CPR method.



(a)

(b)

(c)

Figure 2. Demonstration of a mortar face procedure for a p-interface.
(a) P-interface. (b) Prolongation of the lower order solution (c) Restriction of the common numerical flux

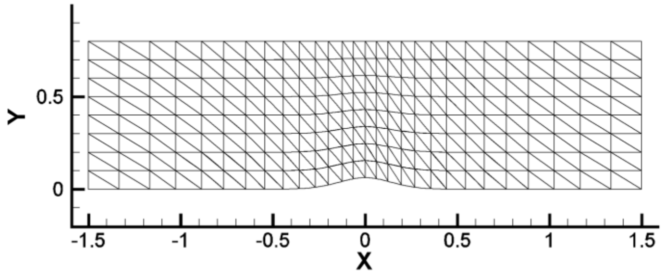


Figure 3. The computational domain and mesh for the subsonic bump flow.

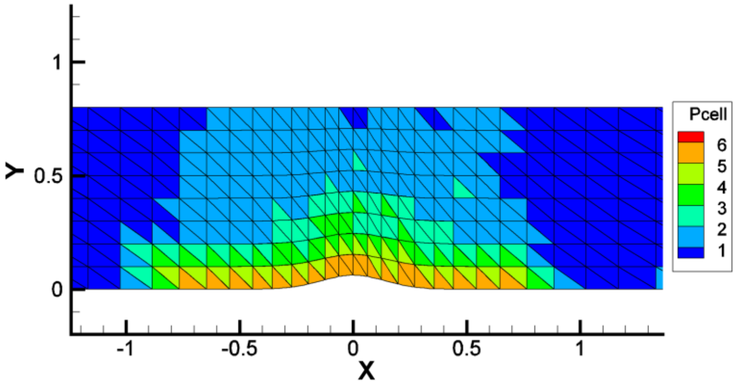


Figure 4. The result of p-adaptation for subsonic bump flow, $f_{\text{adapt}} = 0.2$.

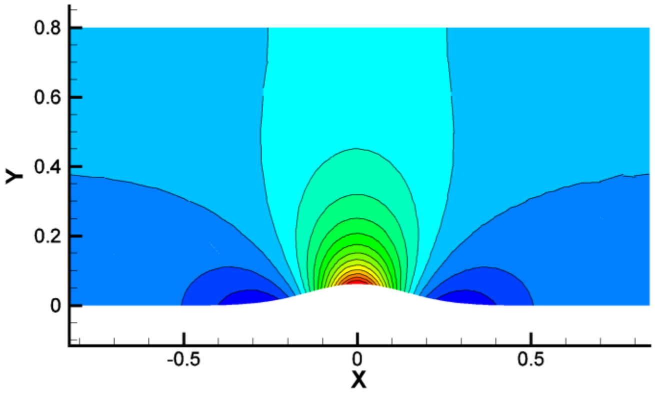


Figure 5. The Ma contours for the bump flow after p-adaptation.

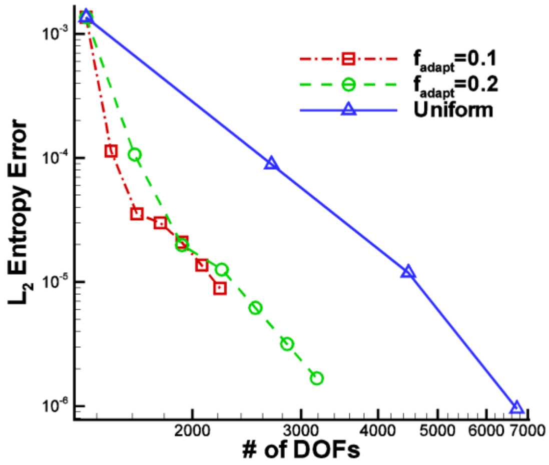
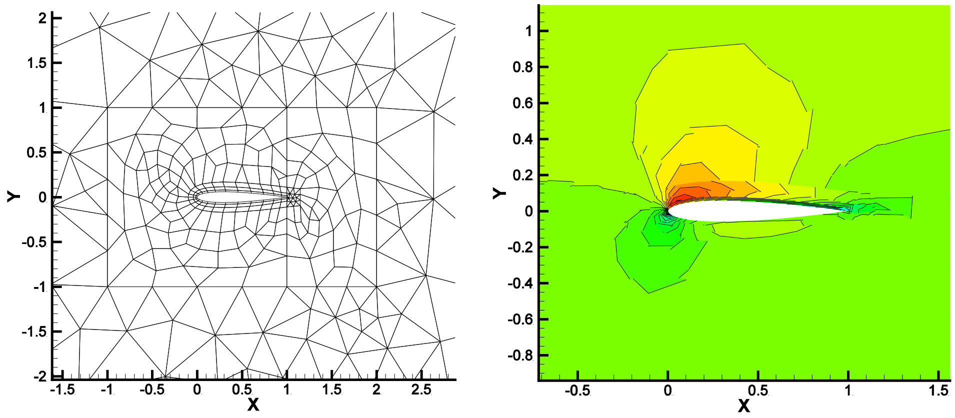


Figure 6. The L_2 entropy error from p-adaptation and uniform p-refinement.



(a)

(b)

Figure 7. (a) Hybrid mesh for subsonic flow over NACA 0012 airfoil. (b) Ma contour of the initial 2nd order solution.

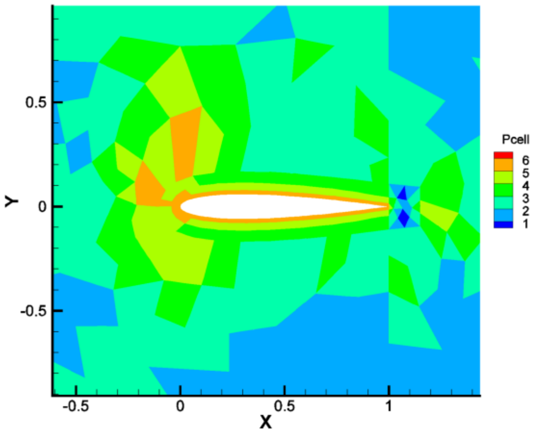


Figure 8. P-adpatation results for the
subsonic flow over NACA0012 airfoil.

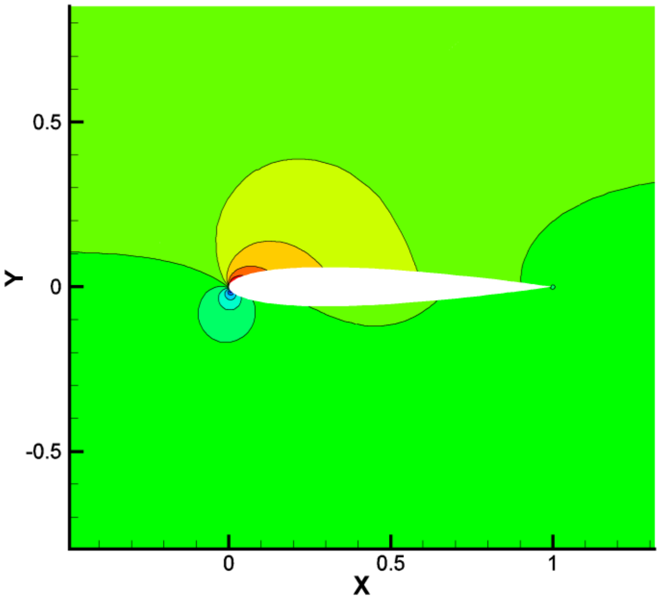
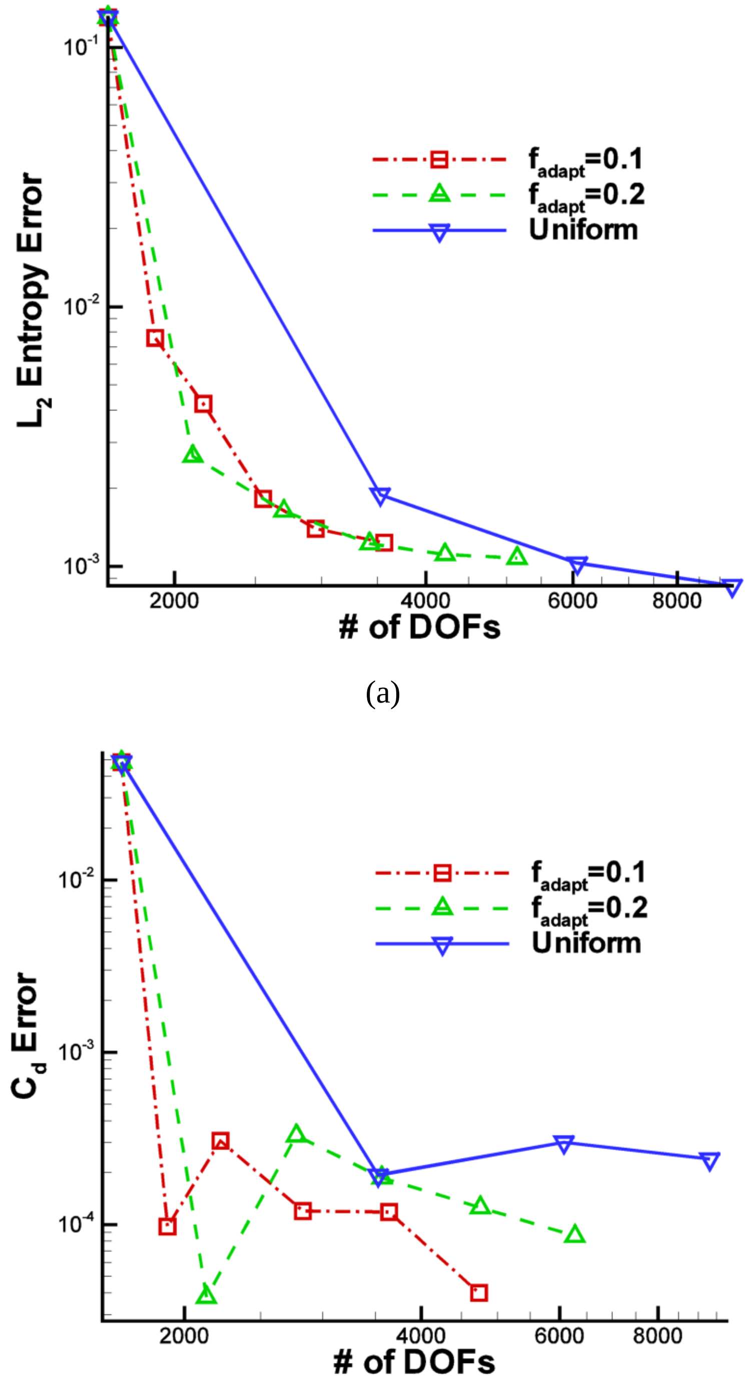


Figure 9. Ma contour of subsonic airfoil flow after p-adaptation.



(a)

(b)

Figure 10 (a) L$_2$ entropy error (b) drag coefficient error for the subsonic airfoil flow.

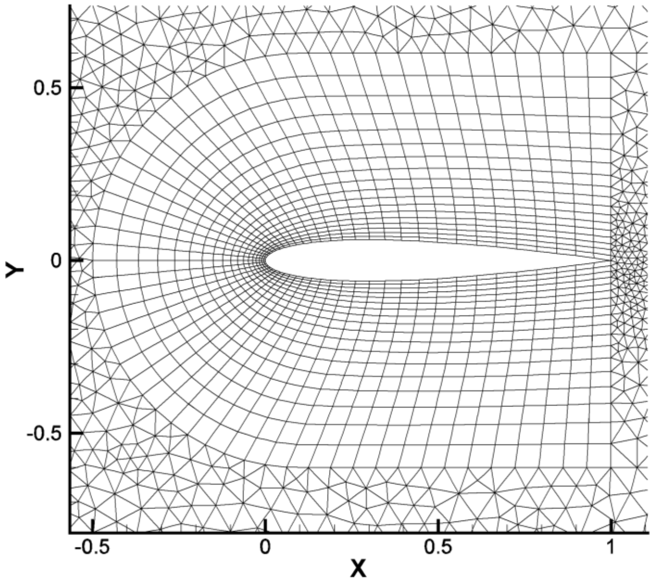


Figure 11. Initial mesh for the transonic airfoil flow.

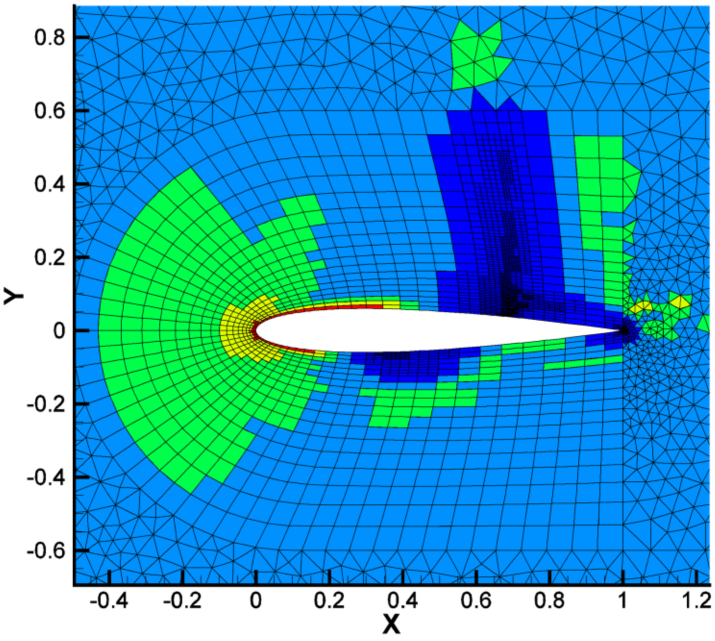


Figure 12. hp-adaptation results for the transonic airfoil flow.

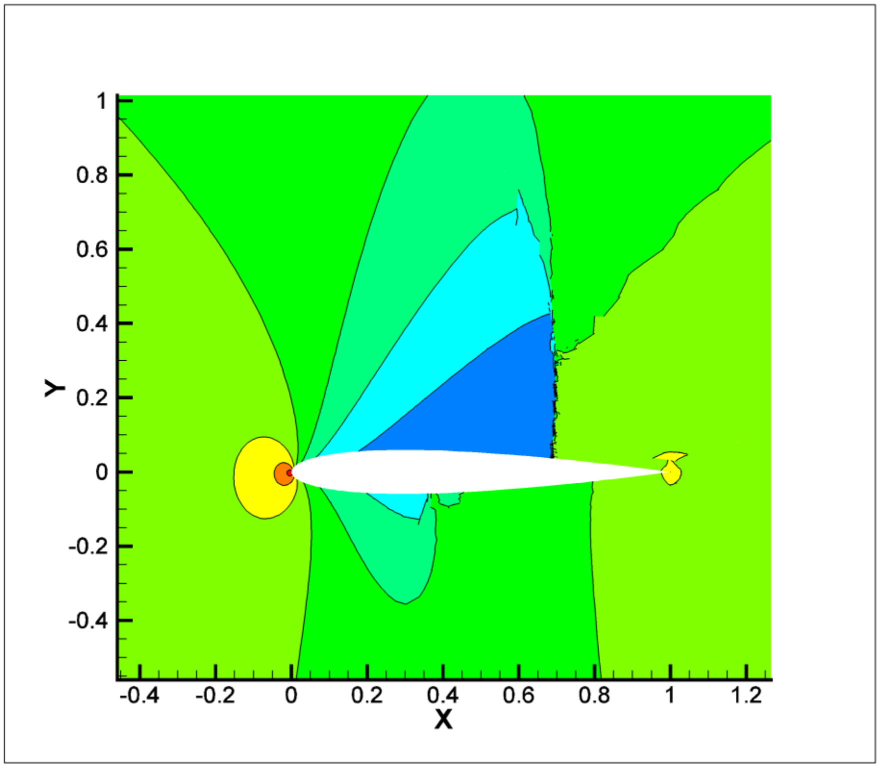


Figure 13. Density contours for the transonic airfoil flow after hp-adaptation.

CHAPTER 6. General Conclusion

A new approach to high-order accuracy, named correction procedures via reconstruction (CPR) is developed for 2D arbitrary mixed mesh in the current work. The CPR formulation combines the accuracy and stability of the popular DG method, and the much improved efficiency via a finite difference like form. The CPR approach is easy to implement with no numerical integration. It is also very flexible in application to hybrid/mixed meshes and development of h/p adaptation capabilities.

The CPR method is successfully implemented for Euler and Navier-Stokes equations. Test results show that it produces optimal accuracy and high-quality numerical solutions, with arbitrarily mixed meshes. A new way of curved boundary representation is also developed and applied to the CPR formulation. The spurious entropy production is shown to be greatly reduced near the curved boundaries, especially with limited geometry information available from today’s mesh generators. The residual-based h/p adaptation procedure offers even more flexibility to the CPR approach, with a rigorous yet easy to compute error indicator. High-quality solution can be achieved with relatively low cost after the adaptation.

Finally, the CPR formulation and related work presented here can be extended to 3D without major obstacles. With its accuracy, efficiency, flexibility and hybrid-mesh capability, the CPR approach shows great promise in solving large-scale flow problems involving complex geometries.

ACKNOWLEDGEMENTS

I would like to take this opportunity to express my thanks to those who helped me with various aspects of conducting research and the writing of this dissertation. First and foremost, Dr. Z.J. Wang his guidance and support throughout this research. I would also like to thank my committee members for their efforts and contributions to this work. Special thanks are also given to my fellow group members for their support and insight provided throughout the years.